Speech recognition based confidence measures for building voices from untranscribed speech

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

MS by Research
in
Electronics and Communication Engineering

by

Tejas Subodh Godambe
201250821
tejas.godambe@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
DECEMBER 2015

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Speech recognition based confidence measures for building voices from untranscribed speech" by Tejas Subodh Godambe, has been carried out under our supervision and is not submitted elsewhere for a degree.

Date

Advisers:
(Dr. Kishore Prahallad & Dr. Suryakanth V Gangashetty)

To my parents, advisers and lab friends

Acknowledgments

First and foremost I thank my adviser Dr. Kishore Prahallad for giving me an opportunity to work with him, and for his continued support. I learnt a lot from seeing how he chooses interesting and impactful research problems to attack and tries to come up with simpler solutions, how he connects different research problems together and his openness and multitasking ability. I feel my approach has been significantly moulded by that. I thank my co-adviser Dr. Suryakanth V Gangashetty for always supporting, giving freedom to work on topic of choice, and not letting any other thing affect my research. I thank Dr. Samudravijaya K from Tata Institute of Fundamental Research, Mumbai for allowing me to leave Mandi project in between for pursuing MS at IIIT-H and for showing faith in me. I thank Prof. B. Yegnanarayana and Prof. Peri Bhaskarrao for exhibiting their undying spirit for fundamental research through their courses and Monday meetings.

I thank all the lab friends in Speech and Vision Lab for being very helpful and making life easy. The names are listed in the order of chronology in which I met them. I thank Vishala, Gautam, Naresh, Mohan, Bhargav, Karthik, Ravi, Arpita, Nivedita, Sivanand, Anandswarup, Santosh, Sudarshan, Raghu, Bhanu, Harsha, Padmini, Patha, Sai Krishna, Mounika, Hari Krishna, A. Raju, V. Raju, Sirisha, Ayushi, Bhavya, and many more.

Lastly, I thank my parents for everything.

Abstract

Today, large amount of audio data is available on the web in the form of audiobooks, podcasts, video lectures, video blogs, news bulletins. In addition, we can effortlessly record and store audio data such as read/lecture/impromptu speech on hand-held devices. These data are rich in prosody, provide a plethora of voices to choose from, and their availability can significantly reduce the overhead of data preparation involved in building general purpose synthesizers, thus helping to rapidly building synthetic voices. But, a few problems such as the following are associated with readily using this data for speech synthesis (1) these audio files are generally long and audio-transcriptions alignment is memory intensive (2) available corresponding transcriptions are approximate, (3) many times no transcriptions are available at all, (4) the audio may contain disfluencies and non-speech noises, since the audio is not specifically recorded for building synthetic voices, and (5) if we obtain automatic transcripts, they are not error free. Earlier works on long audio alignment which addressed the first and second issue generally preferred reasonable transcripts, and mainly focused on (1) less manual intervention, (2) mispronunciation detection and (3) segmentation error recovery. In this thesis, we used a public domain large vocabulary automatic speech recognition (ASR) system to obtain transcripts, followed by confidence measure based data pruning which together address the five issues with the found data, and also ensure the above three points. For proof of concept, we built voices in English language using audiobook (read speech) in female voice downloaded from Librivox and lecture (spontaneous speech) in male voice downloaded from Coursera using both reference and hypotheses transcriptions, and evaluated them in terms of intelligibility and naturalness with the help of perceptual listening test on Blizzard 2013 corpus. The results of subjective intelligibility and naturalness test show that we can build voices of quality comparable to those built using reference transcriptions with the use of automatic transcripts and confidence measure based data pruning.

Keywords: Unit selection synthesis, found data, speech recognition, Librispeech, confidence measures, data pruning

Contents

List of Figures

List of Tables

Abbreviations

AF	Articulatory Features
ASR	Automatic Speech Recognition
CART	Classification and Regression Trees
CM	Confidence Measures
CMU	Carnegie Mellon University
FA	False Acceptance
FR	False Rejection
MCD	Mel Cepstral Distortion
MFCC	Mel Frequency Cepstral Coefficients
MLP	Multilayer Perceptron
MOS	Mean Opinion Score
NSW	Non-Standard Words
PER	Phone Error Rate
SPS	Statistical Sarametric Synthesis
TA	True Acceptance
TR	True Rejection
TTS	Text-to-Speech
WER	Word Error Rate

Chapter 1

Introduction

1.1 Text-to-speech synthesis

Text-to-speech synthesis (TTS) is the artificial production of human speech. A computer system used for this purpose is called a speech computer or speech synthesizer, and can be implemented in software or hardware products. A TTS system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions into speech. The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood clearly.

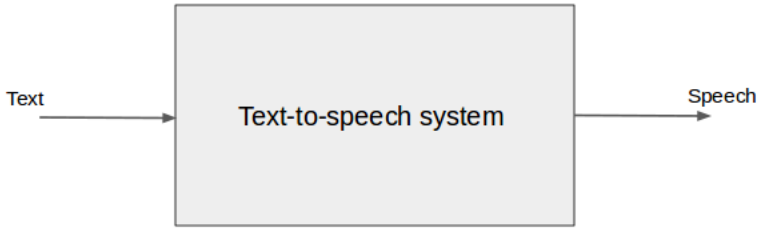

Figure 1.1 A text-to-speech system.

Speech synthesis has several benefits few of which are listed below as cited from http://research.spa.aalto.fi/publications/theses/lemmetty_mst/chap6.html.

- **Applications for the blind**: Today, TTS helps visually impaired people to work with computers, surf internet, listen to spoken form of ebooks etc.

- **Applications for the Deafened and Vocally Handicapped**: People who are born-deaf can not learn to speak properly and people with hearing difficulties have usually speaking difficulties. Synthesized speech gives the deafened and vocally handicapped an opportunity to communicate with people who do not understand the sign language.

With a talking head it is possible to improve the quality of the communication situation even more because the visual information is the most important with the deaf and dumb. A speech synthesis system may also be used with communication over the telephone line [43].

Adjustable voice characteristics are very important in order to achieve individual sounding voice. Users of talking aids may also be very frustrated by an inability to convey emotions, such as happiness, sadness, urgency, or friendliness by voice. Some tools, such as HAMLET (Helpful Automatic Machine for Language and Emotional Talk) have been developed to help users to express their feelings [58]. The HAMLET system is designed to operate on a PC with high quality speech synthesizer, such as DECtalk.

- **Educational Applications**: Synthesized speech can be used also in many educational situations. A computer with speech synthesizer can teach 24 hours a day and 365 days a year. It can be programmed for special tasks like spelling and pronunciation teaching for different languages. It can also be used with interactive educational applications.

  Especially with people who are impaired to read (dyslexics), speech synthesis may be very helpful because especially some children may feel themselves very embarrassing when they have to be helped by a teacher [43]. It is also almost impossible to learn write and read without spoken help. With proper computer software, unsupervised training for these problems is easy and inexpensive to arrange.

  A speech synthesizer connected with word processor is also a helpful aid to proof reading. Many users find it easier to detect grammatical and stylistic problems when listening than reading. Normal misspellings are also easier to detect.

- **Applications for Telecommunications and Multimedia**: The newest applications in speech synthesis are in the area of multimedia. Synthesized speech has been used for decades in all kind of telephone enquiry systems, but the quality has been far from good for common customers. Today, the quality has reached the level that normal customers are adopting it for everyday use.

  Electronic mail has become very usual in last few years. However, it is sometimes impossible to read those E-mail messages when being for example abroad. There may be no proper computer available or some security problems exists. With synthetic speech e-mail messages may be listened to via normal telephone line. Synthesized speech may also be used to speak out short text messages (sms) in mobile phones.

  For totally interactive multimedia applications an automatic speech recognition system is also needed. The automatic recognition of fluent speech is still far away, but the quality of current systems is at least so good that it can be used to give some control commands, such as yes/no, on/off, or ok/cancel.

- **Other Applications and Future Directions**: In principle, speech synthesis may be used in all kinds of human-machine interactions. For example, in warning and alarm systems synthesized

speech may be used to give more accurate information of the current situation. Using speech instead of warning lights or buzzers gives an opportunity to reach the warning signal for example from a different room. Speech synthesizer may also be used to receive some desktop messages from a computer, such as printer activity or received e-mail.

In the future, if speech recognition techniques reach adequate level, synthesized speech may also be used in language interpreters or several other communication systems, such as videophones, videoconferencing, or talking mobile phones. If it is possible to recognize speech, transcribe it into ASCII string, and then resynthesize it back to speech, a large amount of transmission capacity may be saved. With talking mobile phones it is possible to increase the usability considerably for example with visually impaired users or in situations where it is difficult or even dangerous to try to reach the visual information. It is obvious that it is less dangerous to listen than to read the output from mobile phone for example when driving a car.

1.1.1 Components of TTS system

A typical architecture of a Text-to-Speech (TTS) system is as shown in Fig. 1.2. The components of a text-to-speech system could be broadly categorized as text processing and speech generation methods.

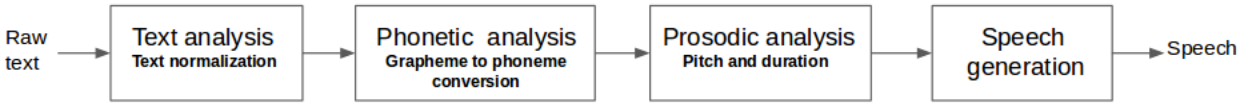

Figure 1.2 Architecture of a text-to-speech system.

1.1.1.1 Text processing

In the real world, the typical input to a text-to-speech system is text as available in electronic documents, news papers, blogs, emails etc. The text available in real world is anything but a sequence of words available in standard dictionary. The text contains several non-standard words such as numbers, abbreviations, homographs and symbols built using punctuation characters such as exclamation !, smileys :-) etc. The goal of text processing module is to process the input text, normalize the non-standard words, predict the prosodic pauses and generate the appropriate phone sequences for each of the words.

Normalization of non-standard words

The text in real world consists of words whose pronunciation is typically not found in dictionaries or lexicons such as IBM, CMU, and MSN etc. Such words are referred to as non-standard words (NSW). The various categories of NSW are: 1) numbers whose pronunciation changes depending on whether

they refer to currency, time, telephone numbers, zip code etc. 2) abbreviations, contractions, acronyms such as ABC, US, approx., Ctrl-C, lb., 3) punctuations 3-4, +/-, and/or, 4) dates, time, units and URLs. Many NSWs are homographs, i.e., words with same written form but different pronunciation. Some of the examples are: 1) IV which may be variously four (Article IV), the fourth (Henry IV), or I.V. (IV drip), 2) three or four digit numbers which could be dates and ordinary numbers (in 2040, 2040 tons). Machine learning models such as Classification and Regression Trees (CART) are used to predict the class of NSW which is typically followed by rules to generate appropriate expansion of a NSW into a standard form [73].

Grapheme-to-phoneme conversion

Given the sequence of words, the next step is to generate a sequence of phones. For languages such as Spanish, Telugu, Kannada, where there is a good correspondence between what is written and what is spoken, a set of simple rules may often suffice. For languages such as English where the relationship between the orthography and pronunciation is complex, a standard pronunciation dictionary such as CMU-DICT is used. To handle unseen words, a grapheme-to-phoneme generator is built using machine learning techniques [9].

Prosodic analysis

Prosodic analysis deals with modeling and generation of appropriate duration and intonation contours for the given text. This is inherently difficult since prosody is absent in text. For example, the sentences where are you going?; where are you GOING? and where are YOU going?, have same text-content but can be uttered with different intonation and duration to convey different meanings. To predict appropriate duration and intonation, the input text needs to be analyzed. This can be performed by a variety of algorithms including simple rules, example-based techniques and machine learning algorithms. The generated duration and intonation contour can be used to manipulate the context-insensitive diphones in diphone based synthesis or to select an appropriate unit in unit selection voices [11].

1.1.1.2 Speech generation

The methods of conversion of phone sequence to speech waveform could be categorized into parametric, concatenative and statistical parametric synthesis.

Parametric synthesis

Parameters such as formants, linear prediction coefficients are extracted from the speech signal of each phone unit. These parameters are modified during synthesis time to incorporate co-articulation and prosody of a natural speech signal. The required modifications are specified in terms of rules which are derived manually from the observations of speech data. These rules include duration, intonation, co-articulation and excitation function. Examples of the early parametric synthesis systems are Klatts

formant synthesis [43] and MITTalk [3].

Concatenative synthesis

Derivation of rules in parametric synthesis is a laborious task. Also, the quality of synthesized speech using traditional parametric synthesis is found to be robotic. This has led to development of concatenative synthesis where the examples of speech units are stored and used during synthesis. The speech units used in concatenative synthesis are typically at diphone level so that the natural co-articulation is retained [59]. An index of the units in the speech database is then created based on the segmentation and acoustic parameters like the fundamental frequency (pitch), duration, position in the syllable, and neighboring phones. At run time, the desired target utterance is created by determining the best chain of candidate units from the database (unit selection). This process is typically achieved using a specially weighted decision tree. The quality of unit selection synthesis is found to be more natural than diphone and parametric synthesis. However, unit selection synthesis lacks the consistency i.e., in terms of variations of the quality [12].

Statistical parametric synthesis

Statistical Parametric Synthesis (SPS) is one of the latest trends in TTS. In contrast to the selection of actual instances of speech from a database in concatenative synthesis, SPS has also grown in popularity over the last few years. SPS might be most simply described as generating the average of some set of similarly sounding speech segments. This contrasts directly with the desire in unit selection to keep the natural unmodified speech units, but using parametric models offers other benefits. The SPS methods offer simplicity in storage by encoding the speech data in terms of a compact set of parameters, and also provide mechanisms for manipulation of prosody, voice conversion etc. The SPS methods are found to produce intelligible and consistent speech as compared to natural and often inconsistent speech by unit selection techniques [84]. Still, the proponents of SPS know that the best examples of SPS are not as good as the best examples of concatenative synthesis. Also the process of reconstructing speech from parameters is still not ideal. Although modeling the spectral and prosodic features is relatively well defined, models of residual/excitation have yet to be fully developed even though composite models such as STRAIGHT [39] are proving to be useful.

1.1.2 Evaluation

In order to evaluate the quality of text-to-speech systems, subjective and objective evaluations are used. In subjective evaluation, the synthetic speech is played to native speakers and their view on the quality of speech is sought. In objective evaluations the synthetic speech is compared with the natural speech utterance and metrics such as spectral distortion are computed. The following sections discuss more about each metrics.

1.1.2.1 Subjective evaluation

Mean opinion score (MOS)

Mean opinion score is probably the most widely used and simplest method to evaluate speech quality in general. It is also suitable for overall evaluation of synthetic speech. MOS is a five level score between 1 (worst) to 5 (best). The listeners task is to evaluate the synthetic speech with scale mentioned in Tale 1.1 below. However, the use of simple five level scale is easy and provides some instant explicit information, the method gives any segmental or selected information on which parts of the synthesis system should be improved [30].

Table 1.1 Scale used in MOS

Scale	Meaning
1	Worst
2	Poor
3	Fair
4	Good
5	Best

AB-Test

AB-Test is also mostly used subjective evaluation for comparing two synthesis techniques. In this method, each listener is subjected to the same sentence synthesized by two different synthesizers are played in random order and the listener is asked to decide which one sounds better for him/her. They also had the choice of giving the decision of equality.

1.1.2.2 Objective evaluation

Mel cepstral distortion (MCD) is an objective error measure used to compute cepstral distortion between original and the synthesized MCEPs. Lesser the MCD value the better is the synthesized speech. MCD is essentially a Euclidean Distance defined as

$$MCD = \frac{10}{ln10} * \sqrt{2 * \sum_{i=1}^{25} (mc_i^r - mc_i^s)^2} \tag{1.1}$$

where mc_i^r and mc_i^s denote the the i^{th} dimension of target and the estimated MCEPs, respectively. MCD is used as an objective evaluation of synthesized speech [8]. Informally it is observed in [8] that an absolute difference of 0.2 in MCD values makes a difference in the perceptual quality of the synthesized signal and typical values for synthesized speech are in the range of 5 to 8.

1.2 Thesis statement

Overhead of data preparation for building general purpose synthetic voices

Building a new general purpose (non-limited domain) unit selection voice in a new language from scratch includes a huge overhead of data preparation, which includes preparing phonetically balanced sentences, recording them from a professional speaker in various speaking styles and emotions in a noise-free environment, and manually segmenting or correcting the automatic segmentation errors. All of it is time consuming, laborious and expensive, and it restricts rapid building of synthetic voices. A free database such as CMU ARCTIC [45] has largely helped to rapidly build synthetic voices in English language. But CMU ARCTIC is a small database, contains only a few speakers data, and it is not prosodically rich (contains short declarative utterances only). Today, (1) large amount of audio data has become available on the web in the form of audiobooks, podcasts, video lectures, video blogs, news bulletins etc, (2) thanks to technology, we can effortlessly record and store large amounts of high quality single speaker audio such as lecture/impromptu/read speech etc. on hand-held devices. Unlike CMU ARCTIC, these data are rich in prosody and provide a plethora of voices to choose from, and their use can significantly ease the overhead of data preparation thus allowing us to rapidly build general purpose natural sounding synthetic voices.

Issues with using found data for building synthetic voices

Now, the question to be asked is whether we can readily use such data to build expressive unit selection synthetic voices [32], and will the synthesis be good? In this thesis, we try to answer this question. There are a few problems related to it such as the following: (1) the audio files are generally long and audio-text alignment becomes memory intensive, (2) precise corresponding transcriptions are unavailable, (3) often no transcriptions are available, and manually transcribing the data from scratch or even correcting the imprecise transcriptions is laborious, time-consuming and expensive, (4) the audio may contain bad acoustic (poorly articulated, dis-fluent, unintelligible, inaudible, clipped, noisy) regions as the audio are not particularly recorded for building TTS systems, and (5) if we obtain automatic transcripts using a speech recognition system, the transcripts won't be error-free.

In case transcriptions are available, long audio alignment techniques could be used to produce smaller chunks of speech and corresponding text, which could then be aligned using Viterbi algorithm to produce phone-level segmentations and consequently used for building TTS systems.

Previous works on long audio alignment

Earlier works have addressed the first and second issue with found data mentioned above in following three ways: (1) audio to audio alignment, (2) acoustic model to audio alignment and (3) text to text alignment. Each method has its advantages and limitations.

1. **Audio-to-audio alignment**: Here, the text is converted to speech using a TTS system and the synthesized speech is aligned with the audio [4, 14]. This method requires existence of a TTS system.

2. **Acoustic model to audio alignment**: Here, acoustic models are aligned with the audio. [Prahallad, 2011] used a modified Viterbi algorithm to segment monologues. Their method assumed a good (atleast 99%) correspondence between speech and text, required manual intervention to insert text at the beginning and endings of monologues, did not handle mispronunciations, and propagated error in one segment to subsequent segments. [Cerisara, 2009] released a Java based GUI to align speech and text. They also used acoustic models, assumed good correspondence between audio and text, and required manual intervention.

3. **Text-to-text alignment**: Here, full-fledged automatic speech recognition (ASR) system including acoustic and language model is used. Basically, long files are chunked into smaller segments based on silence. Hypothesis transcriptions are obtained for these smaller segments. [Moreno, 1998] proposed a method where search is made to see where sequence of words in reference and hypothesis transcriptions match. The stretch where they match are aligned with audio using Viterbi algorithm. This process is repeated until a forced-alignment is done for each audio chunk. The process is practically difficult to implement, and relies on correctness of reference and hypothesis transcriptions. [Moreno, 2009] used finite state transducer based language model instead of N-grams. [Bordel, 2012] used a phone-level acoustic decoder without any phonotactic or language model and then found the best match within the phonetic transcripts. This approach was inspired by the fact that the data to be aligned could have a mixture of languages. But phonetic alignment is less robust than that at word-level. [Braunschweiler, et al., 2009] quantified the number of insertions, substitutions and deletions made by the volunteer who read the book "A Tramp Abroad" by Mark Twain, and proposed a lightly-supervised approach that accounts for these differences between audio and text. Their method, unlike forced-alignment approach in [56] which uses beam pruning to identify erroneous matches, could also find the correct sequence and not only the best match in terms of state sequence between text and audio chunk. [Tao, 2010] proposed a dynamic alignment method to align speech at the sentence-level in the presence of imperfect text data. The drawback of this method is that they cannot handle with phrase reordering within the transcripts.

Above works on long audio alignment addressing the first and second problem with found data generally prefer reasonable transcripts, and mainly focus on (1) less manual intervention, (2) mispronunciation detection and (3) segmentation error recovery.

1.2.1 Contribution of this thesis

In this thesis, we used Librispeech data [60] (which is a large ASR corpus available in public domain) to train a p-norm deep neural network acoustic model [86] and a higher order 4-gram language model having 200,000 unique words. This large vocabulary ASR system was used to obtain automatic transcripts for found data such as an audiobook (read speech) in female voice downloaded from Librivox, and lecture (spontaneous speech) in male voice downloaded from Coursera. Two *reference* ASR systems, one using audiobook data and other using lecture data were also trained. The word and phone error rates of the obtained automatic transcripts provided by the three ASR systems were decent (Table 3.3). To prune the errors made by the ASR system, and the speech/non-speech noises present in the audio (as the audio was not specifically recorded for building TTS systems), posterior probability given by the ASR system was used. In current implementation, we used quinphone, and backoff units such as quadphones, triphones, biphones and monophones for synthesis. Posterior probability of a unit was calulated as minimum of posterior of phones in that unit. All units having posterior probability less than maximum posterior of 1.0 (which gives least false acceptances) were pruned. Posterior probability worked well to harness as much correct hypotheses (number of true acceptances), and retained only a few false alarms (incorrect phone hypotheses of the ASR system which are termed as correct by the posterior probability based confidence measure) (Table 4.6). But, pruning data based on posterior probability does not necessarily prune fast spoken unintelligible words such as common words, short words etc. Neither it necessarily prunes emphasized/hyper-articulated words. Several instances of both short and unnaturally long words were present particularly in the lecture data. So, in addition to posterior probability we used duration as a confidence measure and pruned a unit if its duration was much deviant from the mean duration of units of the same type (having same phone sequence and word position).

For proof of concept, we built voices from both audiobook and lecture speech using automatic transcriptions obtained from both Librispeech ASR and reference ASR system. Thus we built four voices in total. Through subjective intelligibility and naturalness test, we observe that (1) voice of quality comparable to a voice built using transcriptions from reference ASR system, can be built using transcriptions from Librispeech ASR system, (2) Pruning based on posterior probability and duration helps improve intelligibility of speech. It also improves naturalness, but it degrades when more units are pruned.

Through above demonstration, we tried to address the five issues with found data:
(1) We were able to build good voices using found data which were read speech and lecture speech.
(2) We simulated the cases of availability of *approximate* transcripts and also the situation of availability of no transcripts. The reference ASR was trained with approximate transcripts, and its hypothesis was used to build TTS system. Librispeech ASR was used to provide transcriptions when no transcriptions were available.
(3) The fourth and fifth issue such as presence of speech/non-speech noises and wrong labeling by ASR were handled using confidence-measures.

(4) The three important requirements of long audio aligners such as less manual intervention, mispronunciation detection and segmentation error recovery are also taken care of.

1.3 Organization of the thesis

The organization of the thesis is as follows.

- **Chapter 2** gives an overview of automatic speech recognition and confidence measures.
- **Chapter 3** gives an overview of the complete system which takes untranscribed speech as input and outputs a synthetic file. It then describes the data used for building ASR systems, and also the data preparation steps for building TTS systems. Later, it describes the ASR and TTS system development details. Performance of ASR systems trained on audiobook and lecture data, and tested on audiobook and lecture data respectively is compared with performance of ASR system trained with Librispeech data and tested on audiobook and lecture data.
- **Chapter 4** investigates the utility of articulatory features as confidence measures. It summarizes the previous works for pruning spurious units from speech databases, and also the relevance of posterior probability and unit duration as confidence measures for unit pruning. Experiments verifying the effectiveness of posterior probability as a confidence measure, and effect of data pruning based on posterior probability and duration threshold on intelligibility and naturalness of synthetic signal are discussed.
- **Chapter 5** summarizes the work and gives directions for future work.

intro

Chapter 2

Speech recognition based confidence measures

2.1 Automatic speech recognition

Figure 2.1 shows the architecture of a typical speech recognition system. The image was taken from (http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-345-automatic-speech-recognition-spring-2003/).

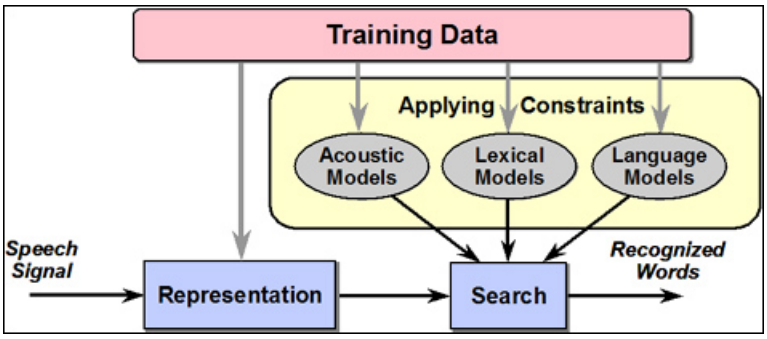

Figure 2.1 Architecture of a speech recognition system.

2.1.1 Statistical pattern classification in ASR

2.1.1.1 Feature extraction

Feature extraction from speech signals is carried out by initially segmenting the raw speech waveforms into frames which are typically 10-20 ms in length, over which the waveforms are assumed to be stationary. Spectral properties of each segment are then calculated in order to yield a low-dimensional representation of the speech segment x. These spectral properties are usually calculated based on some model of human hearing. One common technique makes use of the mel-scale. This scale approximates the manner in which the human auditory system perceives changes in frequencies. In order to extract a representation for the audio, a Fourier transform is applied to the audio signal. The frequency axis in

this spectrum is warped according to the mel scale, before a discrete cosine transform is applied to the mel log powers. The amplitudes in the resulting cepstrum yields the mel frequency cepstral coeffcients (MFCCs) [20].

2.1.1.2 The classification task

Speech recognition may be formulated as a pattern classification task. If the feature vector $X = x_1 \,..\, x_T$ corresponds to the sequence of observation feature vectors extracted from the T frames in the speech waveform, the optimal classifier for the task may be defined as the one

$$W^* = \underset{W}{\operatorname{argmax}} \; P(W|X) \tag{2.1}$$

where the term $P(W|X)$ represents the a-posteriori probability of the word sequence W. This form of classifier will therefore assign the sequence of words W^* with the highest a posteriori probability to the pattern X, and is referred to as the maximum a posteriori probability (MAP) classifier. Equation 2.1 may be expanded with Bayes theorem to yield

$$W^* = \underset{W}{\operatorname{argmax}} \; \frac{P(X|W)P(W)}{P(X)} \propto \underset{W}{\operatorname{argmax}} \; P(X|W)P(W) \tag{2.2}$$

The term $P(X|W)$ in equation 2.2 is the likelihood that the feature sequence X was generated by the underlying acoustic model, and $P(W)$ is the a priori language probability which is given by a language model. The acoustic and language models which are estimated from the training data and subsequently substituted into the above decision rule formula, are explained below. The denominator term $P(X)$ is generally omitted as it is independent of the class, and therefore has no significant bearing on the decision rule. Assuming the distributions $p(X|W)$ and $P(W)$ are the "true" distributions, a classifier which implements the decision rule is guaranteed to minimize the misclassification rate [23].

2.1.2 Components of ASR system

2.1.2.1 Acoustic model

Speech is a time-varying signal. Each word we utter contains a sequence of sounds or phones.
Now,

- Articulation of a phone is a random process, i.e. even if we articulate the same phone, each time its acoustic realization is different, but still similar.
- More precisely, articulation of a phone is a sequence of random processes, where there is transition from the preceding phone to the current phone, steady state and transition to the succeeding phone.
- So, normally multi-state HMMs are used where each state in a multi-state HMM roughly models/represents/characterizes a random process within a phone.

- Traditionally, Gaussian mixture models (GMMs) have been used to model a HMM state. Lately, deep neural networks are being used.

HMMs are trained for each phone in different left and right contexts. Baum-Welch re-estimation algorithm is used for training HMMs. During recognition, these HMMs are evaluated and give the acoustic likelihood $P(X|W)$ in the equation 2.2.

2.1.2.2 Language model

The prior distribution $P(W)$ is represented by a language model (LM). This distribution is purely based on the word sequence W, and in its simplest form may be estimated by obtaining frequency counts from a training corpus of text. Language models which include some word context/history (n-1 words in length) in the estimate, are referred to as *n-gram* LM. These models represent the prior probability of a particular word at index i in a sequence as $P(W_i|W_{i-1}..W_{i-n})$. Some care must however be taken when higher order n-gram models are estimated for a given vocabulary and data size. Increased context lengths yield less reliable n-gram statistics, as the number of representative examples for the more infrequent contexts tails off drastically. Approaches which are typically used to address this issue include the use of models which backoff to lower order n-grams for infrequent contexts, and those which perform statistical smoothing of the n-gram scores.

2.1.2.3 Decoder

Search

Finding the best word sequence for a given sequence of observation vectors was formulated in terms of the MAP criterion in equation 2.2. Using this equation directly represents a search problem. A brute force solution would be to firstly enumerate all possible hypotheses for word sequences. Thereafter, a network of HMMs could be created for each such hypothesis by concatenating the word-level HMMs together. expression in the maximization of equation 2.2 should then be evaluated, would include summing over all possible state sequences in the HMM network to calculate the acoustic model term. The hypothesis under which this likelihood is maximized represents the MAP solution.

In all but the simplest of ASR tasks the aforementioned approach is however computationally in- feasible, and in LVCSR more tractable solutions to the search problem must be sought. A simplifying assumption that may be made is the following:

$$P(X|W) = \sum_{q} p(X, q|W) \approx \max_{q} \; p(X, q|W) \tag{2.3}$$

where the state sequence q is the state sequence through all HMMs in the search network. This is known as the Viterbi Approximation. assumption here is that the likelihood of the best path through the HMM network $\hat{q}$ dominates the sum in equation (2.3), and is therefore an adequate approximation of the sum

term. The search problem is now greatly simplified, as only the best state sequence $\hat{q}$ need be recovered from the HMM network.

The search space may be constrained further in LVCSR tasks by carrying out *hypothesis pruning* (also known as *beam search*). functions in such a manner that at every time instant, the decoder essentially disregards state sequences which fall below a certain likelihood threshold or beam width.

Lattices

A lattice is a graphical structure which is used to represent the pruned hypothesis space which results after running a recognition pass over the acoustic data. The lattice normally includes the hypothesized label (word and/or phone), the corresponding start and end times of the individual linguistic units, as well as the acoustic and language model likelihoods. Structurally, a lattice is essentially a directed, acyclic graph consisting of nodes and edges. Each edge is labeled with the hypothesized linguistic unit, the start and end times of that unit, and the associated acoustic and language model likelihood scores. Each node in the network corresponds to a point in time within the utterance. The identity of the start and end node for each edge is therefore used to impose the required structure. Figure 2.2 is an example of a word lattice for a short utterance. The figure was taken from (http://www1.icsi.berkeley.edu/ suhang/cs267.html.

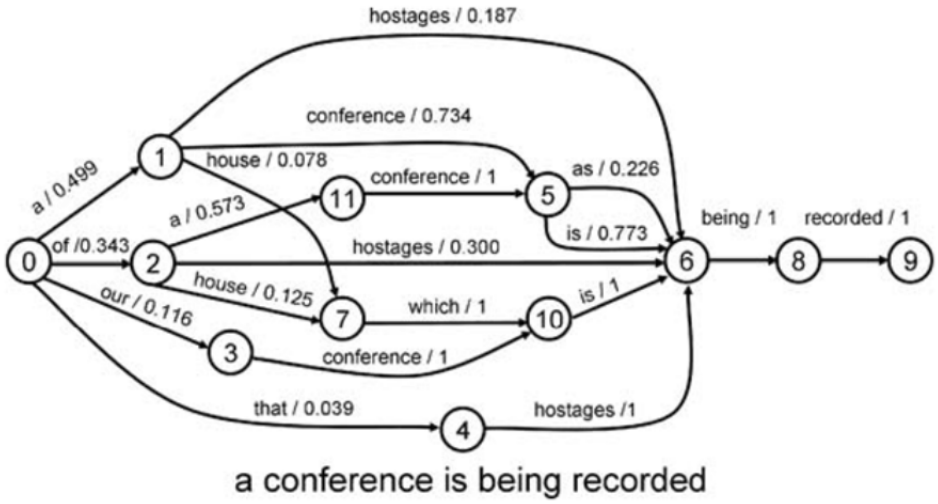

Figure 2.2 Word lattice.

2.1.3 Evaluation

The most commonly used metric in evaluating the performance of LVCSR systems, is the word error rate (WER). This measure is based on the number of words which dier between the hypothesized transcriptions generated by the ASR system, and the reference transcriptions. However, these transcriptions

do not necessarily have the same length. The comparison may therefore not be made by simply comparing the word identities in the two sequences at each index in the reference transcription. The sequences must therefore be aligned using a dynamic programming (DP) procedure. This procedure is carried out so as to minimise the Levenshtein edit distance between the two sequences, which is defined as the weighted sum of occurrences of the following error types:

- Substitution errors, which occur when a reference word is aligned with a hypothesized word which does not match the reference word.
- Deletion errors, which occur when a reference word cannot be aligned with any word in the hypothesis.
- Insertion errors, which occur when an additional word is present in the hypothesis that cannot be aligned with a suitable word in the reference.

The WER may then be calculated as the resulting total number of errors divided by the number of reference words. The WER metric discussed here only considers the best transcription hypothesized by an underlying ASR system (also referred to as the MAP hypothesis or N-Best transcription). Other metrics are also often used in evaluating LVCSR systems. One such metric is the oracle error rate, which is the lowest WER possible over a list of N competing transcriptions (referred to as the N-Best transcriptions), or over an entire lattice.

2.2 Confidence measures

Confidence measures (CMs) are typically used to detect and possibly rectify the incorrect ASR hypothesis. The ASR systems of today, including the most sophisticated ones, fail miserably when migrated from laboratory demonstrations to real-world applications because of ambient noises, speaker variations, channel distortions and several other mismatches. This necessitates the use of a good confidence measure which would compensate for ASR's recognition errors and, in turn, reduce the risk/cost of decisions based on the ASR's output when the ASR is used as part of a larger system, for e.g. in a speech-to-speech translation system, spoken dialogue system etc. In general, confidence measures can be used along with any classifier in any domain unless ground truth is available for verifying classifier's decision and, if going with a wrong decision of the classifier bares moderate to high cost. Other applications of confidence measures in speech domain include detection of non-speech noises, out-of-vocabulary words, detection and correction of human/automatic transcription errors in large training corpus, unsupervised adaptation, selection of reliable (high confidence) speech segments for further processing, etc.

Several confidence measures have been proposed in the past. Please refer to [37] for a survey. According to [37], most of the earlier approaches can be roughly classified into three major categories: (a) CM as a combination of predictor features, (b) CM as posterior probability and (c) CM formulated as utterance verification or statistical hypothesis testing problem. The three formulations of CM are explained below.

2.2.1 The three formulations of CM

2.2.1.1 CM as combination of predictor features

Formulation

The set of features used in classifying word hypotheses are commonly known as predictor features. Many such features have been proposed in the literature, the majority of which are derived from information which is represented in recognition lattices. Theoretically, a predictor feature is any measure for which the distribution over correctly hypothesized words is significantly different from the distribution over incorrectly recognized words. Some of the more common classes of predictor features, along with examples from each class are described below.

1. **Features based on the lattice structure**: Hypothesis Density [40] is a measure based on the assumption that the number of alternative arcs spanning the time segment for a word in the most likely transcription is indicative of the recognizers uncertainty in the hypothesis. This is an intuitive supposition as the recognizer normally prunes word hypotheses from the lattice in situations where the most likely word is significantly more likely than competing words. Word Trellis Stability [69] is a measure based on the premise that a word is more likely to be correct if it is present in a number of competing word hypotheses spanning a similar time interval

2. **Acoustic-based features**: The Acoustic Likelihood Scores normalised per frame or by the number of phones [62], is a natural representation of the match between the acoustic signal and the hypothesized word. Acoustic Stability [85] is a measure based on the number of times a given word occurs in the same (aligned) position in K different outputs from the recognizer, where each of the K outputs is generated using different values of the Grammar Scaling Factor (GSF). This has the effect of weakening the coupling between the language model and acoustic model. Hypothesised words which are often aligned with the same position under different values for are more likely to match the acoustics.

3. **LM features**: A LM feature of particular interest described in [79] is the back-off behaviour (assuming a back-off LM is used by the recognizer). The premise here is that the language model is less confident in a particular word hypothesis when it has had to back-off to a lower order n-gram likelihood.

4. **Duration-related features**: Measures based on the number of phones comprising the hypothesized word, and the duration of the individual phones, may be constructed from phone marked recognition lattices. Such measures were also suggested in [79]. The premise being that shorter phone sequences and words typically correspond to regions of poor performance by the acoustic model.

5. **Word-level utterance features**: Measures such as word posit ion, the length of the utterance, and the lexical identity of the hypothesized word are straight forward to compute. A classifier

which employs such features will be similar in many ways to the Language Model used by the recognizer.

The majority of these predictor features are however not ideal in the sense that there is typically a significant amount of overlap between the resultant distributions for correctly recognized words and incorrectly recognized words. It has therefore widely been posited that the best approach is to combine predictor features in some way, so as to boost their individual discriminative power. Much of the literature which assumes this approach to CM is therefore primarily concerned with investigating different combinations of features, and classification frameworks under which to combine the individual features. A good representative work which explored the definition, combination and subsequent evaluation of a number of predictor features for CM is [16]. Statistical classifiers which have been experimented with for predictor feature combination a re Decision Trees, Maximum Entropy (MaxEnt) models, Generalised Additive Models (GAMs) and Support Vector Machines (SVMs) to name a few.

Shortcomings

Combining sets of predictor features has been shown to improve performance over single-score confidence measures in some cases. However, such approaches may only really be successful in improving performance when the individual predictor features are statistically independent. A study in [40] showed that most predictor features are in fact highly correlated. The result is that attempts at combining features do not yield significant gains over the be st single predictor feature without considerable design effort. The design of a statistical classifier which can successfully combine multiple, arbitrary, potentially highly correlated predictor features to estimate a composite confidence measure is therefore an area of interest.

2.2.1.2 CM as posterior probability

Formulation

Due to the statistical nature of speech recognition systems, it can safely be assumed that the recognizers own belief in the hypotheses it is assigning is a good indication of the true accuracy of the resulting best transcription. This theoretical view point forms the basis of the posterior-based approaches to confidence estimation described in this section. The principles of statistical ASR systems were discussed in Section 2.1.1.2. The MAP decision rule was formulated as choosing the hypothesized word sequence with the maximal posterior probability. This posterior probability may indeed be interpreted as a confidence score. The problem was however reformulated with Bayes Rule to suit a generative HMM-based framework, and the denominator term was disregarded as it played no significant role in the decision rule (equation 2.2). This is a common practice in HMM-based ASR systems, due primarily to the fact that explicitly modeling the distribution $p(X)$ would entail summing over all possible hypotheses. This is an intractable task for large vocabulary systems. The result is however that the posterior distribution is not actually modeled at all. Posterior probabilities must therefore be estimated by either making certain

assumptions about the form of the distribution $p(X)$, or otherwise by employing approximate methods to estimate the distribution explicitly. In the first class of solutions, so-called filler-based methods approach the problem by using a set of simpler, highly constrained filler models to represent the required distribution. Some example techniques are the use of all-phone recognition models [83], catch-all models [38], and making an approximation based on the highest word score assigned by the recognizer [19].

The second broad approach, which aims to employ approximate methods to estimate the true distribution $p(X)$ has however proved more successful. Recognition lattices are compact representations of the most significant competing hypotheses generated by a recognition pass. As such, the hypotheses in a lattice would contribute the majority of mass to the afore-mentioned intractable marginalisation over all hypothesis/word sequences to estimate $p(X)$ directly. Computing posteriors over the lattice is therefore a fairly good approximate method. This does however still represent a computationally intensive task. In [68] and [81] the problem was further simplified by restricting the summation to the N-best hypotheses in the lattice. However, with more elegant algorithms and readily available compute resources - more general approaches which take into account a greater proportion of the hypothesis space from the recognition lattice are feasible. These Lattice-based posterior approaches will be described in more detail in the following sections.

Shortcomings

Lattice-based posteriors typically overestimate the true posterior distribution, as they are computed on a subset of the hypothesis space. They are also susceptible to the independence assumptions made by the recognizer. The raw posteriors must therefore be mapped to confidence scores in some way (as is discussed in [24]). This entails the implementation of an additional post-processing step before the confidence scores can be attached to the transcription. Furthermore, the heuristic nature of the arc clustering and consensus clustering approaches based on overlap and similarity scores is also not completely ideal, and is a potential source of error in the posterior estimation process. As posterior-based approaches have however generally proven to yield better performance than most alternative approaches, research into addressing these shortcomings is warranted.

2.2.1.3 CM as utterance verification

Formulation

Mainly motivated by speaker verification problem,, have proposed to tackle confidence measure problems from a different perspective Under the framework of utterance verification (UV), the confidence measure problem in ASR is formulated as a statistical hypothesis testing problem. For a given speech segment X, assume that an ASR system recognizes it as word W which is represented by an HMM λ_W. Utterance verification is a post-processing stage to examine the reliability of the hypothesized recognition result. Under the framework of UV, we first propose two complementary hypotheses, namely the

null hypothesis H_0 and the alternative hypothesis H_1 as follows:

H_0: X is correctly recognized and truly comes from model λ_W
H_1: X is wrongly classified and is NOT from model λ_W

Then we test H_0 against H_1 to determine whether we should accept the recognition result or reject it. According to NeymanPearson Lemma, under some conditions, the optimal solution to the above testing is based on a likelihood ratio testing (LRT), i.e.,

$$LRT = \frac{p(X|H_0)}{p(X|H_1)} \gtrless \tau \tag{2.4}$$

where τ is the critical decision threshold. The LRT-based utterance verification provides a good theoretical formulation to address confidence measure problems in ASR. The above LRT score can be transformed to a confidence measure based on a monotonic one-to-one mapping function. The major difficulty with LRT is how to model the alternative hypothesis which usually represents a very complex and composite event, where the true distribution of data is unknown. In practice, the same HMM structure is adopted to model the alternative hypothesis, which can be a general background model, or hypothesis-specific anti-model, or a set of competing models, or a combination of all the above.

Shortcomings
A significant challenge in UV-based CM is that of estimating the alternate models, as has already been mentioned. Such models represent highly complex composite distributions, and there is no clearly defined way to define these alternatives to the null hypothesis models. A truly alternative model may for instance be trained on different data, or may be based on a completely different modelling technique. This has been an ever-present challenge discussed in the literature on UV-based CE techniques, and is yet to be resolved.

2.2.2 Evaluation

When evaluating confidence measure annotation, we usually encounter two types of errors, namely false alarm errors and false rejection errors. Obviously, receiver operating characteristic (ROC) curve gives a full picture of verification performance at all operating points. In many cases, it is convenient to use a single-number metric for CM assessment. Some widely used metrics include equal error rate (EER), confidence error rate, normalized cross entropy, etc. Refer to [40], [72], [53] and [82] for details. Another important issue in CM evaluation is to take recognition boundaries into account. For example, a correctly recognized word may have a very low confidence measure because its boundary is wrong (though its identity is correct). Thus, it is helpful to use the concept of word-correctness proposed by [79] in evaluating CMs.

Chapter 3

System development

3.1 System overview

Figure 3.1 shows the architecture of the entire system to which input is untranscribed speech data and whose output is the synthesized audio file. First, the ASR system accepts the audio data and produces corresponding labels. Then, data pruning using confidence measures takes place. The unpruned audio and label data form the unit inventory for the TTS system. During synthesis time, the TTS system accepts normalized text, takes into account the duration and phrase break information predicted by a statistical parametric speech synthesizer trained using the same audio data and hypothesized transcriptions, and chooses an appropriate sequence of units that minimize the total of target and concatenation costs. The output of the TTS system is an audio file.

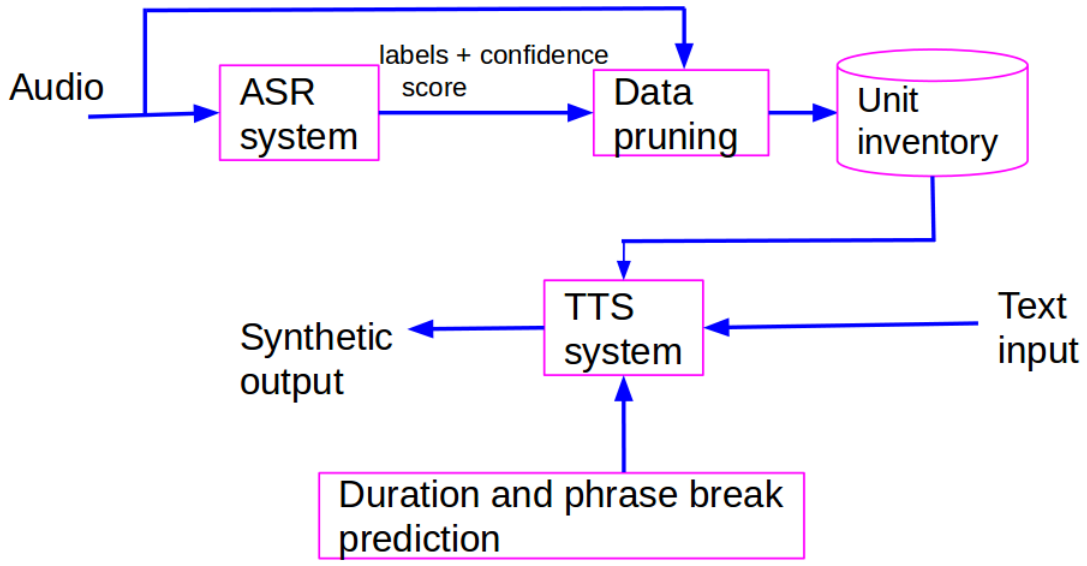

Figure 3.1 Architectural block diagram of the complete system for building unit selection voices from untranscribed speech.

3.2 Data preparation

3.2.1 Data used for building ASR systems

We built three ASR systems, one of which was built using Librispeech data [60]. Librispeech is a fairly recently made available continuous speech corpus in English language, which is prepared by collating parts of several audiobooks available at Librivox website. It contains two parts: 460 hours of clean speech and 500 hrs of speech data containing artificially added noise. We used 460 hrs of clean speech[1] to build acoustic models, and a 3-gram language model[2] pruned with threshold $3\text{x}10^{-7}$ to generate the lattices, and a higher order 4-gram language model[3] to rescore the lattices and find the 1-best Viterbi path for the ASR system.

The other two ASR systems (both acoustic and language models) were built using the TTS data described in the next subsection.

3.2.2 Data preparation for building TTS system

Details of the audio used for building voices are given in Table 3.1 and Table 3.2 below. One is an audiobook (read speech) in female voice downloaded from Librivox and other is a lecture (spontaneous speech) in male voice downloaded from Coursera. The audio files were converted to 16kHz WAV format and power normalized. Before downloading the audio, we checked that the voice quality and speech intelligibility of the speakers are good, and that the audio has not been recorded in a noisy background. For the audiobook, we also made sure that it is not a part of the 460 hrs clean speech Librispeech corpus which was used for training ASR system so that we can simulate the situation that the found audiobook data is unseen by the ASR system. For the lecture speech, a few lectures contained TED talks and voices from other speakers, both of which were removed. The audiobook and lecture audio files were long, and could not be directly used for decoding as memory shortage problems can arise while running Viterbi algorithm. So, silence based chunking of the long audio files is usually performed. In this work, however, we obtained audio chunks using open source tool Interslice [64] as we also wanted to obtain chunks of corresponding reference transcripts for building TTS system using reference transcripts for comparing it with TTS system built using hypothesis transcripts. The start and end of each spoken chapter of audiobooks generally contains metadata such as "This is a Librivox recording" and reader's name which is not present in text chapters downloaded from Project Gutenberg. Since Interslice requires agreement between text and speech, we manually checked and added/deleted text at the start and end of each chapter. Same process was carried out even for lecture speech. Since Interslice does not have a mechanism to prevent propagation of segmentation error to subsequent segments, we also manually

[1] http://www.openslr.org/12/

[2] http://www.openslr.org/resources/11/3-gram.pruned.3e-7.arpa.gz

[3] http://www.openslr.org/resources/11/4-gram.arpa.gz

verified the agreement between start and end of resulting speech and text chunks before using them for building TTS systems.

Table 3.1 Details of the audiobook used for building unit selection voice.

Name of the audiobook	**Author**	**Read by**	**Running time**
Olive (Voice 1)	Dinah Maria Mulock CRAIK	Arielle Lipshaw	14:03:13

Table 3.2 Details of the lecture speech used for building unit selection voice.

Name of the course	**Instructor**	**Running time**
Introduction to Public Speaking (Voice 2)	Dr. Matt McGarrity	$\approx$ 12hrs

3.3 ASR and TTS system

3.3.1 ASR system development details

Lately, ASR systems have become much more accurate and robust thanks to deep neural networks (DNNs) [61, 75, 57]. We used scripts provided with Kaldi toolkit [63] for training DNN-based ASR systems, and IRSTLM tool [25] for building language models. Kaldi is based upon finite state transducers, and it is compiled against the OpenFst toolkit [2].

3.3.1.1 Acoustic modeling

Figure 3.2 shows the flow of the steps followed for training acoustic models.

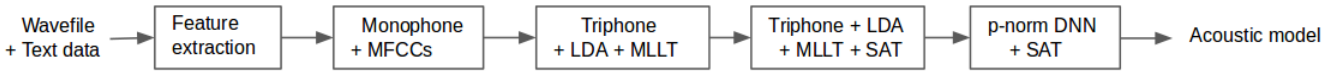

Figure 3.2 Steps followed for training acoustic model.

1. **Feature extraction**: First, 13 dimensional Mel frequency cepstral coefficients (MFCCs) [20] were extracted. Hamming window of 25 ms frame size and 10 ms frame shift was used. Then, cepstral mean subtraction is applied on a per-speaker basis. Then MFCCs were appended with the velocity and acceleration coefficients.

2. **Training monophone system**: A set of context-independent or monophone Gaussian mixture model-hidden Markov model (GMM-HMM) acoustic models were trained on above features.

3. **Training triphone system with LDA+MLLT** : MFCCs without the deltas and acceleration coefficients were spliced in time taking a context size of seven frames (i.e. ± 3). These features, were de-correlated and their dimensionality was reduced to 40 using linear discriminant analysis (LDA) [23]. Further de-correlation was applied on resulting features using maximum likelihood linear transform (MLLT) [31] which is also known as global semi-tied covariance (STC) [27].

4. **Training triphone system with LDA+MLLT+SAT**: Then speaker normalization was applied on above features using feature-space maximum likelihood linear regression (fMLLR), also known as constrained MLLR (CMLLR) [26]. The fMLLR was estimated using the GMM based system applying speaker adaptive training (SAT) [54, 26].

5. **Training DNN system**: A DNN-HMM system with p-norm nonlinearities [86] was trained on top of SAT features. Here GMM likelihoods are replaced with the quasi-likelihoods obtained from DNN posteriors by dividing them by priors of the triphone HMM states.

3.3.1.2 Lexicon

Lexicon was prepared from most frequent 200,000 words in the Librispeech corpus. Pronunciations for around one-third of them were obtained from CMUdict. The pronunciations for the remaining words were generated using the Sequitur G2P toolkit [7].

3.3.1.3 Language modeling

We used the IRSTLM toolkit [25] for training language models. Modified Kneser-Ney smoothing was used [44, 17].

3.3.1.4 Decoding

First, hypothesis transcriptions were produced using the spliced MFCC features. These transcriptions were then used to estimate the fMLLR transforms as explained above. The accuracy of the hypothesis transcriptions obtained after SAT was much better than that before SAT.
Decoding of the audiobooks was done in two passes. In the first pass, a relatively computationally inexpensive language model was used to generate a lattice or word-graph of competing alternative hypotheses. In the second pass, a computationally expensive and higher order language model was used to re-score the language model probabilities on the lattice, re-rank the alternative hypotheses and find the 1-best hypothesis.
Even though we required phone labels for the audio for building the TTS system, direct phone decoding

was not performed as it normally leads to high errors. Rather, word decoding was performed first, and then word lattices were converted to phone lattices using the lexicon lookup.

3.3.2 Experiment: Checking the performance of ASR systems

We trained p-norm DNN-HMM acoustic model for all three ASR systems trained on Olive, lecture and Librispeech data. While decoding Olive and lecture audio, using corresponding ASR systems, we used 3-gram language model prepared on corresponding text. While decoding Olive and Lecture data using Librispeech ASR system, we performed decoding in two passes. In the first pass, a pruned language model having threshold $3\text{x}10^{-7}$ was used to generate the lattices. These lattices were then re-scored with higher order 4-gram language model. The two pass decoding approach was adopted while decoding with Librispeech ASR system alone because Librispeech language model is prepared on large text data and is heavy, and decoding with it can be computationally expensive and time-consuming. Table 3.3 shows the word error rates (WERs) and phone error rates (PERs) given by ASR systems trained on Olive and lecture data, and tested on Olive and lecture data (TTS data) respectively. It also shows the WER and PER of ASR system trained with Librispeech data, and tested on Olive and lecture data. Note, for computing WERs for both Olive and lecture data, we respectively used the word-level transcriptions available at Project Gutenberg website and those available at Coursera, as reference transcriptions. These transcriptions are reliable, *but not gold-standard*. For computing the PERs, phone-level reference transcriptions were obtained by converting word-level reference transcriptions to phones using lexicon lookup. CMU lexicon containing 200,000 words provided with Kaldi setup was used for that purpose. We can observe the following things in Table 3.3

1. As expected, the performance of ASR system trained with Librispeech data is relatively poor but it is still quite decent.

2. The performance gap between ASR system trained on lecture data and Librispeech is big as compared to that between ASR system trained on Olive and Librispeech data because Librispeech is a read speech corpus just like Olive, while lecture data is spontaneous data.

3. The performance of ASR system trained on lecture data and tested on lecture data is slightly poorer than ASR system trained on Olive data and tested on Olive data. The reasons could be that there is relatively more uniformity in Olive data (read speech) compared to lecture speech. Lecture speech is more spontaneous, and contains a lot of filled pauses, dis-fluencies, fast spoken (at times unintelligible words) and also emphasized words.

3.3.3 TTS system development details

For synthesis, we made modifications to the TTS system submitted to Blizzard challenge 2015 [67].

Table 3.3 WERs and PERs of ASR systems trained on TTS data and Librispeech data.

Training data for ASR	**Test data to ASR**	**WER (%)**	**PER (%)**
Olive	Olive	2.54	1.02
Librispeech	Olive	3.74	1.57
Lecture	Lecture	5.91	5.19
Librispeech	Lecture	21.93	10.20

3.3.3.1 Feature extraction and unit inventory preparation

1. **Unit size**: Units of different sizes ranging from frame-sized units [33, 50], HMM-state sized [22, 35], half-phones [6], diphones [12] to syllables [42] to much larger and non-uniform units [71] have been investigated in the literature. Since we were using reasonably large data for synthesis, we chose longer non-uniform units such as quinphones, quadphones, triphones, biphones and monophones. Quadphones, triphones, biphones and monphones serve as back-off units when required quinphones are not available in the data. Use of longer contextual units facilitates fewer joins, and hence fewer possible discontinuities.

2. **Acoustic feature extraction**: Log-energy, 13 dimensional MFCCs, fundamental frequency (F_0) were extracted for every wave file. Frame size of 20 ms and 5 ms frame shift was used. F_0 were extracted using STRAIGHT tool [39]. The durations and posterior probabilities of the phones which we use as confidence measure were obtained from Kaldi decoder.

3. **Preparing the catalog file**: A catalog or dictionary file (which is basically a text file) was prepared which contained the list of all units (including monphone to quinphone) and the attributes of each unit such as duration, start and end times, the duration zscore of each unit type, F_0, log-energy, MFCC values of boundary frames and posterior probability scores (computed as the minimum of posterior probabilities of phones in that unit). This file was used during synthesis time to compute target and join costs explained below.

4. **Pre-clustering units**: Pre-clustering is a method that allows the target cost to be effectively pre-calculated. Typically, units of the same type (phones, diphones etc.) are clustered based on acoustic differences using decision trees [12]. In this work, we clustered units of a type (i.e. units containing the same sequence of phones) on the basis of their positions in words (such as beginning, internal, ending, singleton), as such clustering implicitly considers acoustic similarity. Units in a cluster are typically called as "candidate" units.

3.3.3.2 Steps followed during synthesis time

Figure 3.3 shows the flow of the steps described below.

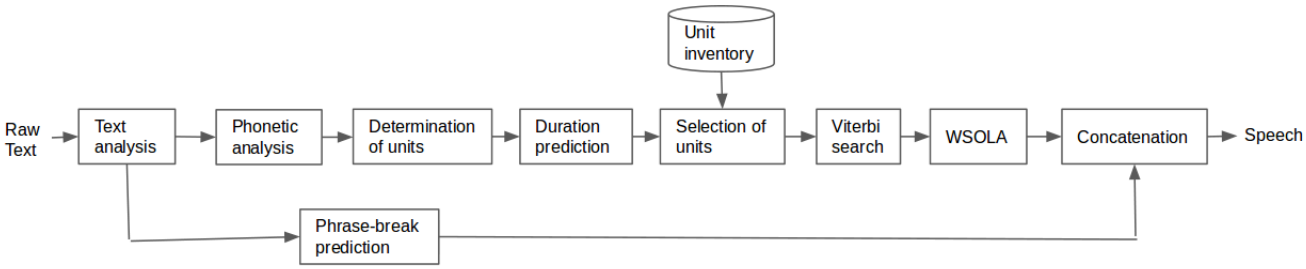

Figure 3.3 Steps followed during synthesis time.

1. **Text normalization**: A test sentence was first tokenized, punctuations were removed, non-standard words such as time, date etc. were normalized, and abbreviations and acronyms were converted to full-forms.

2. **Phonetic analysis**: Each word was broken into sequence of phones using lexicon. Grapheme to phoneme converter [7] was used to convert out-of-vocabulary words and proper names into phone sequence.

3. **Prediction of phrase-break locations**: Using the audio and automatic transcriptions obtained from the ASR systems, we built statistical parametric voices using Clustergen synthesizer [8] in Festival framework [10, 74]. In the current implementation, we took help of the phrase-break locations predicted using classification and regression trees (CART) [13] in Clustergen. For each text input, we first synthesized a statistical parametric voice. A pause unit of appropriate duration was placed at the predicted phrase-break locations during concatenation.

4. **Determination of units**: For every word in the test sentence, we first searched for units of maximum length which are quinphones or units of length equal to length of the word if the word comprised of fewer than five phones. If units of maximum length were not found, then we searched for units of (maximum length-1), and so on. In short we joined units of maximum length when they were available in the database, or used back-off units of shorter lengths. This approach resulted in fewer joins, and a more natural and faster synthesis.

5. **Predicting durations of units**: In the current implementation, we used CART based duration prediction module in Clustergen. For each text input, we first synthesized a statistical parametric voice and used the predicted phone/word durations to select units close to the predicted durations.

6. **Selection of units**:

   - **Target cost computation**: Target cost indicates how close a database unit is to the desired unit. The difference between duration predicted by the CART module in Clustergen [8] and duration of candidate units in database was used as the target cost.

- **Join cost computation**: Join cost indicates how well the two adjacently selected units join together. The join cost between two adjacent units u_{i-1} and u_i was calculated using the following equation, which is a linear weighted combination of distance between log energy, fundamental frequency F_0 (extracted using STRAIGHT tool) and MFCCs of frames near the joining of u_{i-1} and u_i. In the following equation, the symbols α, β and γ respectively denote the weights for log energy, F_0 and MFCC.

$$Join\_cost = \alpha C_{F_0}(u_{i-1}, u_i) + \beta C_{log\_energy}(u_{i-1}, u_i) + \gamma C_{MFCC}(u_{i-1}, u_i) \quad (3.1)$$

  Following [66, 67], we used four context frames while computing distance between log energies and F_0 of u_{i-1} and u_i, as it helped minimize perceived discontinuities.

- **Viterbi search**: The equation 3.2 below explains the way the total cost is computed. The term *Tdist(U_i)* is the difference between duration of unit U_i and the predicted duration, and the term $Jdist(Ui, Ui-1)$ is the join cost of the optimal coupling point between candidate unit U_i and the previous candidate unit it is to be joined to. W_1 and W_2 denote the weights given to target and join costs respectively. N denotes the number of units to be concatenated to synthesize the sentence in question. We then used a Viterbi search to find the optimal path through candidate units that minimized the total cost which is the sum total of target and concatenation costs.

$$\text{Total cost} = \sum_{i=1}^{N} W_1 Tdist(U_i) + W_2 Jdist(U_i, U_{i-1}) \quad (3.2)$$

7. **Waveform similarity overlap addition (WSOLA)**: We used an overlap addition based approach for smoothing the join at the boundaries. Specifically, the cross correlation formulation of WSOLA [76] was used. The algorithm was reformulated in order to first find a suitable temporal point for concatenating the units at the boundary. This ensured that the concatenation is performed at a point where maximal similarity exists between the units. In different words, this ensured that sufficient signal continuity exists at the concatenation point. For this, cross correlation between the units was used as a measure of similarity between the units. Next, the units were joined at the point of maximal correlation using cross-fade technique [33] which further helped remove the phase discontinuities. The number of frames used to calculate the correlation was limited by the duration of the available subword unit. In the current framework, we used the last two frames of the individual units to calculate the cross correlation.

Chapter 4

Data pruning using confidence measures

In almost all techniques belonging to the three categories mentioned in section 2.2.1, the confidence values are predicted by ASR, and the disadvantage is that these confidence measures may become too recognizer specific at times [18]. Naturally, we would like to use a reliable external (ASR independent) information source for CM which we could use independently or in tandem with posterior probability (which is the winner among three approaches for CM) for data pruning. We chose to test usefulness of articulatory features (AFs) for this task. AFs come with a number of important properties, for e.g., AFs are human-readable, they have been successfully applied for robust speech recognition [41] and as tandem features [15]. In addition, AFs have been long researched for speech inversion, and were fairly recently also applied to LVCSR [55]. So, AFs have been proven to be an efficient alternative representation of speech signals and this motivated us to experiment their usefulness for confidence measures. Figure 4.1 shows the architectural flowchart of our approach. The ASR output is verified by checking its degree of compliance with the output of a trained multilayer perceptron (MLP), in the AF space. Very few earlier works have explored AFs for the task of CMs [48, 49].

4.1 Using articulatory feature based confidence measures

Table 4.1 shows the 28 AFs we used in this study, and cardinality of each feature group.
An MLP was trained to map the 13 Mel Frequency Cepstral Coefficients (MFCCs) and their velocity and acceleration coefficients to the 28 AF values. TIMIT data [28] was used for the experimentation purpose. The architecture of the MLP trained was 39L 100N 100N 28S, where L, N and S correspond to linear, tangential and sigmoidal activation functions respectively. This architecture ensured that there is a ratio of atleast 10:1 between the number of training samples (MFCC frames) and number of weights in the MLP so that the MLP weights obtain good generalization. Kaldi toolkit [25] was used for training the ASR system using subspace Gaussian mixture model [26] and MFCC features. Figure 4.2 illustrates, with an example, the flow in Figure 4.1. The initial part of the TIMIT sentence “she said sharks have no bones and shrimp swam backward” was chosen for the illustration purpose. The subplots below, in the top-down order, respectively show AF streams for reference transcription, ASR output, MLP

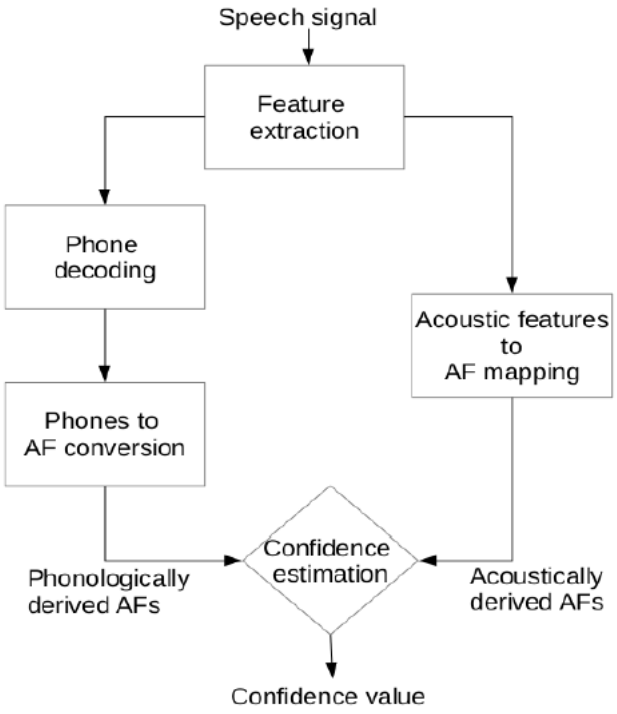

Figure 4.1 Architectural flowchart of the AF-based CM approach.

output, projection of ASR output on MLP output and frame confidence. AF streams in the first and second subplot were obtained by phonologically mapping reference and hypothesis phones to AF values, whereas AF stream in the third subplot was obtained from acoustics by feeding MFCCs as input to the MLP. Note the reference and hypothesis phone labels above the first two subplots. The frame confidence values in the final subplot were calculated as magnitude (inner product) of each frame in the fourth subplot.

4.1.0.3 Preliminary observations

(a) ASR could not distinguish between the confusable sound pairs (eh, ih) and (sh, ch) and committed two substitution errors.

(b) ASR hypothesized the first three phones “sil sh iy” correctly, but their boundaries deviate largely from the reference.

(c) MLP output in the third subplot appears quite similar to the reference. Its hypothesized boundaries look good. But, we can notice a few problems too. Still, it gives a hope that if high amounts of data are used for training MLP as in [15], MLP shall be able to approximate reference even better, and then MLP output shall do a much better job acting as a pseudo reference in the absence of actual reference, for verifying ASR output.

Table 4.1 Articulatory features.

Feature group	**Feature class**	**Cardinality**
Voicing	$\pm voice$, silence	3
Manner	vowel, lateral, nasal, fricative, approximant, silence	7
Place	dental, coronal, labial, retroflex, velar, glottal, high, mid, low, silence	10
Front-back	front, back, nil, silence	4
Rounding	$\pm round$, nil, silence	4

(d) The valleys in the final subplot can be explained with the help of two reasons discussed in point (d).

4.1.0.4 Computing phone confidence

A phone is tagged 'correct' when its confidence value exceeds a threshold, which is generally set at Equal Error Rate (EER, the threshold value at which false alarm rate equals false rejection rate). Three different methods were tested for computing phone confidence from frame confidences (in final subplot in Figure 4.2). (a) In the first method, an average of confidence values of all frames belonging to a phone segment (hypothesized by the ASR) was used as phone confidence. The EER obtained was 41.8%. (b) Using above method, a large difference was observed between the average phone confidence values of stop sounds and other sounds such as fricatives, vowels etc. Stop sounds, in particular, were heavily suffering from false rejections. This meant that, for some reason, confidence values of most stop sounds were not been able to cross the global threshold obtained at EER. Manual inspection of the projected AF streams (fourth subplot in Figure 4.2) of stop sounds showed that, in most cases, except one frame all other frames exhibited low frame confidence. The one frame (possibly the one capturing the stop burst) did exhibit high frame confidence. When we were taking an average of all frames in a segment belonging to a stop sound, that one high confidence frame was getting shadowed by the neighboring low confidence frames. So, we modified the above phone confidence computation technique to regard confidence of the most confident frame in a phone segment as the phone confidence. This modification alleviated the false rejections of stop sounds to a certain extent and also brought a whopping drop of 6% in EER. The new EER obtained was 35.8%. (c) Inspired by the above improvement and, with the knowledge that articulators move asynchronously during production of speech, we decided to consider sum of maximum values of the projected AFs distributed over adjacent frames for phone confidence computation, rather than considering one frame with maximum confidence as that one frame may not contain max. values of all its AFs. This method yielded EER=37.2% (better than first method but worse than second method).

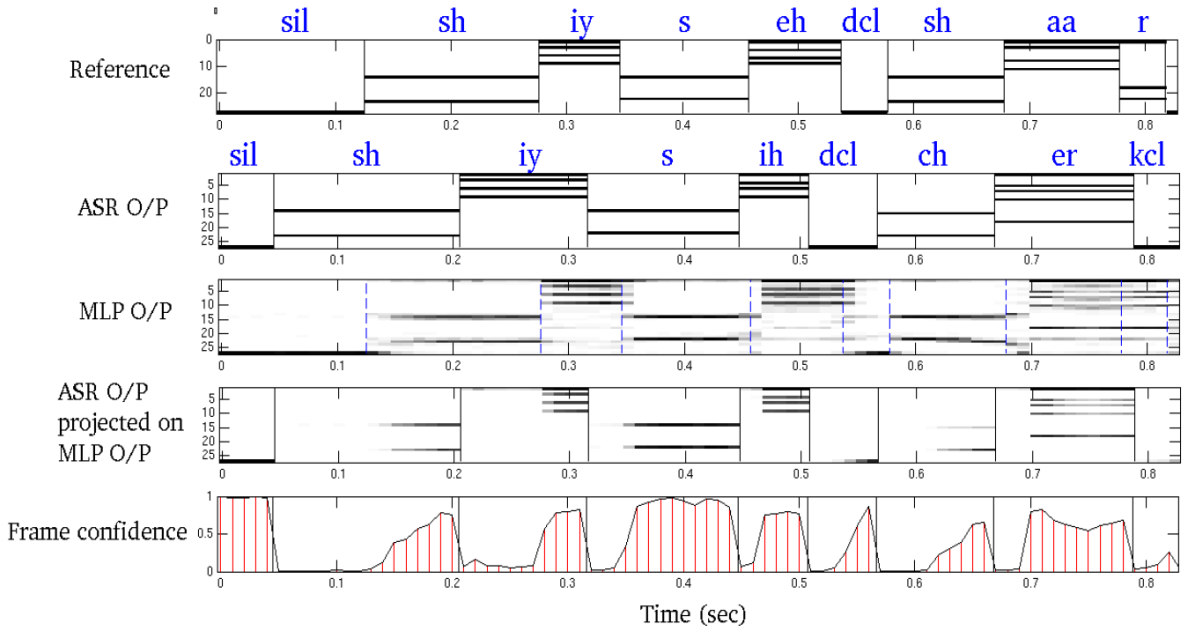

Figure 4.2 AF streams for (a) reference, (b) ASR output, (c) MLP output, (d) projection of ASR output on MLP output, (e) frame confidence.

Table 4.2 EERs for different methods tested to compute phone confidence.

Phone confidence equal to	**EERs (%)**
Average of confidences of frames in the segment	41.8
Maximum of confidences of frames in the segment	35.8
Summing max. values of phone's AFs in the segment	37.2

4.1.0.5 Inspecting component AF values

Table 4.3 shows the average phone confidence for five correctly hypothesized phones by ASR and average values of their projected AFs. AF values are accompanied with names of the AF classes. The threshold value obtained (at EER) was 0.86. By glancing at the AF values we can make out that most instances of "b" and "w" must not have been able to cross 0.86 threshold and must have got falsely rejected and, the low AF values are be the ones to be blamed as they have resulted in low frame confidence and consequently low phone confidence values. Why is there a difference between average voicing values of "b" and "w"? We speculate that degrees of voicing differ for "b" and "w". The sound "w" may be more voiced sound than "b". But, when we trained MLP we did not give different weights to "b" and "w" for voicing. We did not use actual AFs obtained from measurement. We used pseudo AFs obtained by phonological mapping of phones to AFs. Still, MLP learned the different degrees of

Table 4.3 Average AF values for five correct hypothesis of ASR.

Phone	Avg. phone confidence	Voicing	MoA	PoA	Tongue position	Rounding
b	0.72	0.78, voiced	0.56, stop	0.4, labial	0.85, nil	0.70, nil
sh	0.94	0.91, unvoiced	0.98, fricative	0.87, high	0.98, nil	0.98, nil
ng	0.72	0.93, voiced	0.78, nasal	0.21, velar	0.86 nil	0.86, nil
w	0.80	0.96, voiced	0.62, approximant	0.65, labial	0.86, nil	0.86, nil
ay	0.90	0.99, voiced	0.97, vowel	0.83, low	0.82, front	0.94, unbounded

Table 4.4 Comparison of EERs of AF-based and standard CM approach.

CM technique	EERs (%)
Using posterior probabilities	25.6
Using attempted AF-based method	35.8

voicing. It seems that MLP has captured the degree of voicing and, learned that "w" is more voiced that "b". Knowing this, when we apply a general confidence threshold for all phones, it appears to be a sub-optimal decision. We would have to scale the AFs for each phone or use some different features instead of MFCC for training the MLP, or apply some technique that would bring all the phones on the same plane if we still want to stick to applying one threshold for all phones. Another observation was that the place of articulation (PoA) is most difficult to classify. One of the reasons can be that it has maximum number of classes among all feature groups and hence more ambiguity. Second reason could be that MFCC does not seem to carry information that can help discriminate between different PoA classes. Alternative features such as LPCCs, FDLP etc. should be investigated.

4.2 Comparison with the posterior probability approach

It sounds logical to formulate as well as interpret confidence score as probability value as in [29]. Posterior probabilities are typically used for the task as the definition of posterior probability matches with that confidence measures where both try to give an estimate of how likely the ASR hypothesis is correct. Posterior probability of a hypothesized phone given the acoustics can be estimated by any technique such as neural networks, classification and regression trees etc, but those estimated from word graphs (prepared during recognition) have shown better results at discriminating between correct and incorrect ASR hypothesis [80, 82], and have also outperformed the techniques belonging to first and third categories of CMs. One of the reasons behind success of posteriors derived from word graph is that the language model information is implicitly considered along with the acoustic information. Maximum a posteriori method is typically used for recognizing speech where a sentence (phone sequence in our case) with the maximum posterior probability is chosen as the hypothesis. Posterior of a single phone/word in the hypothesized sequence is computed by summing over posteriors of all hypotheses that contain this phone

Table 4.5 Difference between average confidence values of correct and incorrect hypotheses of each phone using both CM approaches.

Phone	C1	C2	Phone	C1	C2	Phone	C1	C2	Phone	C1	C2
b	0.01	0.19	sh	0.00	0.27	m	0.02	0.05	eh	0.01	0.08
d	0.04	0.20	z	0.07	0.28	n	0.12	0.25	ey	0.01	0.24
g	0.04	0.33	f	0.04	0.09	ng	0.13	0.18	ae	0.03	0.14
p	0.13	0.23	th	0.00	0.42	l	0.17	0.14	aa	0.06	0.16
t	0.07	0.21	v	0.06	0.25	r	0.08	0.14	aw	0.04	0.28
k	0.08	0.20	dh	0.08	0.19	w	0.16	0.21	ay	0.05	0.19
dx	0.01	0.17	oy	0.05	0.23	y	0.08	0.19	ah	0.03	0.25
jh	0.03	0.22	uh	0.08	0.03	hh	0.03	0.33	ow	0.07	0.22
ch	0.02	0.18	er	0.10	0.17	iy	0.04	0.10	uw	0.06	0.16
s	0.01	0.14	sil	0.29	0.16	ih	0.03	0.12			

in around the same time region. In this way the phone becomes independent of the phone sequences on either sides, and its posterior probability is obtained. Forward-backward algorithm is used to compute the posterior probability. The drawback of the posterior-based CM approach is that it is non progressive, and word graphs of different sizes usually result in CMs with similar performance [37].

As can be seen in Table 4.4, our attempted approach falls behind the standard posterior probability approach by a large margin. It indicates that we may have to involve a lot more sophistication into our current approach. Since EER indicates the capability of a CM to discriminate between correct and incorrect phone hypothesis we computed, for analysis purpose, the difference between the average confidence values of correct and incorrect hypothesis for each phone. This was done for both standard and attempted CM approach. The results can be seen in Table 4.5. The first, fourth, seventh and tenth columns list the 39 TIMIT phones we used. The second, fifth, eighth and eleventh columns tabulate difference between average confidence values of correct and incorrect hypotheses using our AF-based CM approach. The third, sixth, ninth and twelfth columns tabulate the same but for posterior-based CM approach. We can note that posterior-based CM is much better at discriminating between correct and incorrect hypotheses, which also implicitly explains the large difference between the two EERs in Table 4.4. Two odd cases were also observed. Note values such as -0.01 on the intersection of “b” and “C1”, and -0.03 on the intersection of “jh” and “C2”. In these two cases it happened that average confidence values for incorrect hypotheses were greater than that for correct hypotheses.

4.3 Data pruning

In the context of TTS systems, data pruning involves removal of spurious units (which may be a result of mislabeling or bad acoustics) and units that are redundant in terms of prosodic and phonetic features. Pruning spurious units improves TTS output [12, 34, 51, 1, 78, 46] while pruning redundant

units reduces database size thus enabling portability [52, 47, 65] and real-time concatenative synthesis [70, 21, 36]. In this thesis, we focus on removing spurious units and not redundant units.

4.3.1 Previous works to prune spurious units

In [12] each unit is represented as a sequence of MFCC vectors, and clustering using decision tree proceeds based on questions related to prosodic and phonetic context; each unit is then assessed for its frame-based distance to cluster center. Units which lie far from their cluster centers are termed as outliers and hence pruned. In [34], this evaluation is done on the basis of an HMM framework: only instances which have the highest HMM scores are retained to represent a cluster of similar units. Confidence features such as log-likelihood ratio [51], transcription confidence ratio [1], generalized posterior probability [78] have also been used for pruning. We used posterior probability [82, 78] and unit duration [46] obtained from the ASR system as confidence measures.

4.3.2 Relevance of posterior probability and unit duration as confidence features

Posterior probability helps detecting mislabeled and bad acoustic regions, while unit durational measure helps detecting unnaturally short or long units (which may have high posterior probability values) but can make words unintelligible or sound hyper-articulated respectively. Thus both these confidence features are directly related to intelligibility and naturalness of speech.

4.3.3 Other advantages of posterior probability

Other motivations to use posterior probability as confidence feature are: (1) Posterior probability, by definition, tells the correctness or confidence of a classification, (2) it has been shown to work consistently better than other two formulations of confidence measures, which are confidence measure as a combination of predictor features, and confidence measures posed as an utterance verification problem [37], and (3) posterior probability becomes more reliable when robust acoustic [86] and language models are used (as in this case) [77].

4.3.4 Computation of posterior probability and unit durational zscore

In an ASR system, the posterior probability of a phone or a word hypothesis w given a sequence of acoustic feature vectors $O_1^T = O_1 O_2 .. O_T$ is computed (as given in equation 4.1) as the sum of posterior probabilities of all paths passing through w (in around same time region) in the lattice. It is computed using forward-backward algorithm over the lattice. In the equation below, W_s and W_e respectively indicate sequence of words preceding and succeeding w in a path in the lattice.

$p(w|O_1^T) =$

$$
\begin{aligned}
&= \sum_{W_s} \sum_{W_e} p(W_s w W_e | O_1^T) \\
&= \frac{\sum_{W_s} \sum_{W_e} \left[p(O_1^{t_s} | W_s) p(O_{t_s}^{t_e} | w) p(O_{t_e}^T | W_e) p(W') \right]}{p(O_1^T)} \\
&= \frac{\sum_{W_s} \sum_{W_e} \left[p(O_1^{t_s} | W_s) p(O_{t_s}^{t_e} | w) p(O_{t_e}^T | W_e) p(W') \right]}{\sum_W \left[\sum_{W_s} \sum_{W_e} \left[p(O_1^{t_s} | W_s) p(O_{t_s}^{t_e} | w) p(O_{t_e}^T | W_e) p(W') \right] \right]}
\end{aligned}
\tag{4.1}
$$

Normally, the phone hypotheses in the neighbourhood of a low confidence phone are also affected. Hence, we discard all units containing even a single phone below a specified threshold. The posterior probability of a unit is calculated as the minimum of posterior probability of phones in that unit.

The unit durational zscore for every unit class is computed as in the following equation.

$$
zscore = \frac{duration - mean}{standard\ deviation} \tag{4.2}
$$

4.4 Experimental studies

4.4.1 Experiment 1: Checking the effectiveness of posterior probability as confidence measure

We saw that ASR system trained with Librispeech data produces more accurate and reasonably accurate transcripts for Olive and lecture data respectively. The incorrect phone hypotheses should not be a part of the unit inventory and need to be automatically removed to prevent them from corrupting a synthesized voice. We used posterior probability given by the ASR system as a confidence measure to prune the erroneous data, where all phones below an optimal posterior probability threshold were pruned. In this experiment, we see to what extent our confidence measure is useful to automatically detect bad acoustics and incorrect hypotheses of ASR system. Table 4.6 shows the PER as in Table 3.3 and its breakup in terms of percentage substitution, insertion and deletion errors. Note that we can have posterior probabilities only for hypothesis produced by the ASR system. The phone hypotheses could be correct phones or substitutions or insertions. So, deletions could be detected by the confidence measure, but as their amount was small, we ignored them. The confidence measure is expected to Truly Reject (TR) as many substitution and insertion errors. Table 4.6 also shows the percentage of true acceptances (TA), false rejections (FR), true rejections (TR) and false acceptances (FA) obtained at the maximum and optimal posterior probability threshold value equal to 1.0 for all the four cases in Table 3.3. This threshold value is optimal in the sense that it yields the least number of false acceptances (which are the ASR system's erroneous phone hypotheses termed as correct and hence left unpruned by the confidence measure). We would want the least number of erroneous hypotheses/spurious phones, and hence we prefer least number of false acceptances. We observe that the posterior probability does a decent job to

harness most of the correct data (as can be seen from the percentage of TAs) leaving just a small amount of erroneous data behind (as can be seen from the amount of false acceptances). Specifically, in the case of of lecture speech recognized by ASR system trained with Librispeech data, we can see that 10.20% PER is effectively reduced to (2.77+1.63)% (percentage of FAs) with the use of confidence measure. There is also a sizeable amount of false rejections that we can see but we could afford to lose that data since we were using large data for synthesis.

Table 4.6 Performance of ASR systems and posterior probability confidence measure.

Training data for ASR	**Test data to ASR**	**%PER**	**%sub**	**%ins**	**%del**	**%FA**	**%TR**	**%FR**	**%TA**
Olive	Olive	1.02	0.20	0.66	0.16	0.74	0.12	0.54	98.60
Librispeech	Olive	1.57	0.58	0.69	0.30	1.00	0.27	4.94	93.79
Lecture	Lecture	5.19	0.96	3.52	0.71	3.09	1.39	3.28	92.24
Librispeech	Lecture	10.20	3.27	5.30	1.63	2.77	5.80	19.61	71.82

4.4.2 Experiment 2: Checking the effect of pruning based on posterior probability and unit duration on WER and MOS

There are several examples of fast unintelligible speech (in case of common and short words such as “to”, “the”, “for”, “and”, etc. plus other short words) and unnaturally long or emphasized/hyper-articulated words particularly in lecture speech. Several instances of such words have posterior probability value equal to 1.0, and are left unpruned. Hence, we also prune the units which are much deviant from their mean duration. In addition, pruning units based on duration allows us to prune many more units than it is possible using posterior probability alone.

Table 4.7 shows, for all four voices, the different posterior probability and duration zscore thresholds used to achieve different amounts of unit pruning. The first number in every cell is the posterior probability threshold which is 1.0. The second entry in second row indicates the number of units having posterior probability equal to 1.0. No duration threshold was applied in this case. The second entry in third and fourth rows indicate the duration thresholds applied.

We used the hypotheses and the timestamps given by the ASR systems trained with Olive, lecture and Librispeech data for synthesis. Even in case of Olive and lecture respectively, we used hypotheses of ASR system instead of force-aligned reference transcriptions from Project Gutenberg and Coursera because the reference transcriptions are reliable, but not gold-standard, and the ASR system trained and adapted to a single speaker generally gives better transcriptions, and is able to detect the inconsistencies in reference speech and text.

Table 4.7 Posterior probability and duration zscore thresholds used to achieve different amounts of data pruning, for all four voices.

Percent units used	Posterior probability & duration zscore thresholds			
	Test data = Olive		Test data = Intro. to Public Speaking	
	ASR trained on Olive data	ASR trained on Librispeech	ASR trained on Lecture	ASR trained on Librispeech
100	-	-	-	-
*	1.00, $\approx 97\%$	1.00, $\approx 92\%$	1.00, $\approx 93\%$	1.00, $\approx 65\%$
50	1.00, ± 0.51	1.00, ± 0.70	1.00, ± 0.57	1.00, ± 0.98
30	1.00, ± 0.35	1.00, ± 0.45	1.00, ± 0.39	1.00, ± 0.60

The above table contains 16 different combinations of posterior probability and duration zscore thresholds. We synthesized (for each combination in the table) 10 semantically unpredictable sentences (SUS) [5] and 10 news sentences from the Blizzard 2013 test corpus. So, 20 sentences were synthesized for each combination. In all 320 sentences were synthesized. A few of the samples used for this experiment can be listened to at [1]. These sentences were randomly distributed among 16 listeners for perceptual test. So each listener transcribed 10 SUS (from which we computed the WER indicating the speech intelligibility) and rated the naturalness of the news utterances on a scale of 1 (worst) to 5 (best) from which we calculated the mean opinion score (MOS).

Tables 4.8 and 4.9 respectively show the WER and MOS for all four voices synthesized using different amounts of pruned data. We can see that the WERs and MOS are quite good for all the four voices. We can observe the following things in Table 4.8.

1. The WERs are high for lecture speech than audiobook speech.

2. The WERs are high for unpruned data (as can be seen in the first row). They become slightly better in the second row when we use only units having posterior probability value equal to 1.0 to synthesize the voices. The improvement is maximum in the last column where the amount of pruned data having posterior probability less than 1.0 is the highest.

3. Selecting units close to mean duration (as in the third row) decreases WER even further, as short units which are much deviant from the mean duration are pruned.

4. The improvement in WER observed with duration pruning (difference in WERs of 2nd and 3rd row) is more than the difference in WERs of 1st and 2nd row observed with pruning units having posterior probability less than 1.0. This difference is more evident in case of lecture speech (which contains more units corresponding to fast speech than audiobook contributing to less intelligible speech). The WER further reduces when more units based on duration are pruned (even when

[1] https://researchweb.iiit.ac.in/~tejas.godambe/EURASIP/

only 30% units are retained).

In the case of naturalness of speech in Table 4.9, we can observe the following things.

1. Voices built using audiobook seem to be more natural than those built using lecture speech.
2. The MOS is almost same for first and second rows except the case of last column where noticeable improvement is observed in MOS.
3. The MOS decreases as we move down rows as it becomes difficult to find units having duration close to predicted duration and which can also maintain continuity in terms of energy, F_0 and MFCCs.

Table 4.8 Word error rates for all four voices for different amounts of data pruning.

Percentage units used	**Word error rate (%)**			
	Test data = Olive		**Test data = Lecture**	
	ASR trained on Olive	**ASR trained on Librispeech**	**ASR trained on Lecture**	**ASR trained on Librispeech**
100	14.25	17.21	22.13	28.17
*	13.50	15.95	20.56	23.90
50	8.11	9.56	16.25	17.15
30	6.26	6.28	13.25	13.87

Table 4.9 MOS scores for all four voices for different amounts of data pruning.

Percentage units used	**Mean opinion score**			
	Test data = Olive		**Test data = Lecture**	
	ASR trained on Olive	**ASR trained on Librispeech**	**ASR trained on Lecture**	**ASR trained on Librispeech**
100	3.49	3.52	3.18	2.91
*	3.51	3.47	3.21	3.22
50	3.28	3.22	2.99	3.05
30	3.08	3.00	2.90	2.93

Chapter 5

Summary and Conclusions

Today, large amount of audio data is available on the web in the form of audiobooks, podcasts, video lectures, video blogs, news bulletins etc. In addition, we can effortlessly record and store audio data such as read/lecture/impromptu speech etc. on hand-held devices. These data are rich in prosody, provide a plethora of voices to choose from, and their availability can significantly reduce the overhead of data preparation involved in building general purpose synthesizers, thus helping to rapidly building synthetic voices. But, a few problems such as the following are associated with readily using this data for speech synthesis (1) these audio files are generally long and audio-transcriptions alignment is memory intensive (2) precise corresponding transcriptions are unavailable, (3) many times no transcriptions are available at all, (4) the audio may contain dis-fluencies and non-speech noises, since the audio is not specifically recorded for building synthetic voices, and (5) if we obtain automatic transcripts, they are not error free. Earlier works on long audio alignment which addressed the first and second issue generally preferred reasonable transcripts, and mainly focused on (1) less manual intervention, (2) mispronunciation detection and (3) segmentation error recovery. In this thesis, we tried to address above issues with building synthetic voices from found data in the following way.

We used Librispeech data [60] (which is a large ASR corpus available in public domain) to train a p-norm deep neural network acoustic model [86] and a higher order 4-gram language model having 200,000 unique words. This large vocabulary ASR system was used to obtain automatic transcripts for found data such as an audiobook (read speech) in female voice downloaded from Librivox, and lecture (spontaneous speech) in male voice downloaded from Coursera. Two *reference* ASR systems, one using audiobook data and other using lecture data were also trained. The word and phone error rates of the obtained automatic transcripts provided by the three ASR systems were decent (Table 3.3). To prune the errors made by the ASR system, and the speech/non-speech noises present in the audio (as the audio was not specifically recorded for building TTS systems), posterior probability given by the ASR system was used. In current implementation, we used quinphone, and backoff units such as quadphones, triphones, biphones and monophones for synthesis. Posterior probability of a unit was calulated as minimum of posterior of phones in that unit. All units having posterior probability less than maximum posterior of 1.0 (which gives least false acceptances) were pruned. Posterior probability worked well to harness as

much correct hypotheses (number of true acceptances), and retained only a few false alarms (incorrect phone hypotheses of the ASR system which are termed as correct by the posterior probability based confidence measure) (Table 4.6). But, pruning data based on posterior probability does not necessarily prune fast spoken unintelligible words such as common words, short words etc. Neither it necessarily prunes emphasized/hyper-articulated words. Several instances of both short and unnaturally long words were present particularly in the lecture data. So, in addition to posterior probability we used duration as a confidence measure and pruned a unit if its duration was much deviant from the mean duration of units of the same type (having same phone sequence and word position).

For proof of concept, we built voices from both audiobook and lecture speech using transcriptions obtained from both Librispeech ASR and reference ASR system. Thus we built four voices in total. Through subjective intelligibility and naturalness test, we observe that (1) voice of quality comparable to a voice built using transcriptions from reference ASR system, can be built using transcriptions from Librispeech ASR system, (2) Pruning based on posterior probability and duration helps improve intelligibility of speech. It also improves naturalness, but it degrades when more units are pruned.

Through above demonstration, we tried to address the five issues with found data:
(1) We were able to build good voices using found data which were read speech and lecture speech. (2) We simulated the cases of availability of *approximate* transcripts and also the situation of availability of no transcripts. The reference ASR was trained with approximate transcripts, and its hypothesis was used to build TTS system. Librispeech ASR was used to provide transcriptions when no transcriptions were available. (3) Issues such as presence of speech/non-speech noises and wrong labeling by ASR were handled using confidence-measures.

The three important requirements of long audio aligners such as less manual intervention, mispronunciation detection and segmentation error recovery are also taken care of.

Future Works

1. Need better techniques or combination of techniques to (*with very high accuracy*) automatically detect and remove all audio segments unworthy and damaging for the final synthesized voice.

2. With that, the audio data recorded in even more realistic/noisy conditions can be thought to be useful for readily building synthetic voices.

3. Need to incorporate beam search to synthesize long sentences and paragraphs in real-time.

4. In the present implementation, we used large amount of data (more than 10 hrs) for synthesis. We wish to re-conduct the experiment and see how well it fares with small amount of data (with less than or equal to 4 hrs).

Publications

Publications related to the thesis

1. **Tejas Godambe**, Sai Krishna Rallabandi, Suryakanth V Gangashetty, Ashraf Alkhairy and Afshan Jafri, *Developing a unit selection voice given audio without corresponding text*, EURASIP Journal of Audio and Music Processing.

2. **Tejas Godambe**, Sai Krishna Rallabandi and Suryakanth V Gangashetty, *Data pruning and objective assessment of intelligibility using confidence measures for unit selection synthesis system*, XRCI Open 2016.

Other publications

1. Sai Krishna R, Ayushi Pandey, Sai Sirisha R, **Tejas Godambe** and Suryakanth V Gangashetty, *Sonority rise: Aiding back off in syllables-based synthesis*, Twenty Second National Conference on Communications (NCC), 2016.

2. Sivanand Achanta, **Tejas Godambe** and Suryakanth V Gangashetty, *An Investigation of Recurrent Neural Network Architectures for Statistical Parametric Speech Synthesis*, Interspeech 2015.

3. **Tejas Godambe**, Nandini Bondale, Samudravijaya K and Preeti Rao, *Multi-speaker, narrowband, continuous Marathi speech database*, Oriental COCOSDA held jointly with IEEE International Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE), pp. 1-6, IEEE, 2013.

4. **Tejas Godambe**, Namrata Karkera and Samudravijaya K, *Adaptation of acoustic models for improved Marathi speech recognition*, Proc. of 1st International Symposium on Acoustics, November 2013, New Delhi, India.

5. **Tejas Godambe** and Samudravijaya K, *Speech data acquisition for voice based agricultural information retrieval*, Proc. of 39th All India DLA Conference, Punjabi University, Patiala, June 2011.

Bibliography

[1] J. Adell, P. D. Agüero, and A. Bonafonte. Database pruning for unsupervised building of text-to-speech voices. In *Proc. ICASSP*, volume 1, pp. I–I, 2006.

[2] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFst: A general and efficient weighted finite-state transducer library. In *Implementation and Application of Automata*, pp. 11–23. Springer, 2007.

[3] J. Allen, M. S. Hunnicutt, D. H. Klatt, R. C. Armstrong, and D. B. Pisoni. *From Text to Speech: The MITalk system*. Cambridge University Press, 1987.

[4] X. Anguera, N. Perez, A. Urruela, and N. Oliver. Automatic synchronization of electronic and audio books via TTS alignment and silence filtering. In *Proc. ICME*, pp. 1–6. IEEE, 2011.

[5] C. Benoît, M. Grice, and V. Hazan. The SUS test: A method for the assessment of text-to-speech synthesis intelligibility using semantically unpredictable sentences. *Speech Communication*, 18(4):381–392, 1996.

[6] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, and A. Syrdal. The AT&T next-gen TTS system. In *Joint meeting of ASA, EAA, and DAGA*, pp. 18–24. Citeseer, 1999.

[7] M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451, 2008.

[8] A. W. Black. CLUSTERGEN: a statistical parametric synthesizer using trajectory modeling. In *Proc. Interspeech*, 2006.

[9] A. W. Black, K. Lenzo, and V. Pagel. Issues in building general Letter to Sound Rules. 1998.

[10] A. W. Black, P. Taylor, R. Caley, R. Clark, K. Richmond, S. King, V. Strom, and H. Zen. The festival speech synthesis system, version 1.4.2. *Unpublished document available via http://www.cstr.ed.ac.uk/projects/festival.html*, 2001.

[11] A. W. Black and P. A. Taylor. Assigning intonation elements and prosodic phrasing for English speech synthesis from high level linguistic input. 1994.

[12] A. W. Black and P. A. Taylor. Automatically clustering similar units for unit selection in speech synthesis. 1997.

[13] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression trees*. CRC press, 1984.

[14] N. Campbell. Autolabelling Japanese TOBI. In *Proc. ICSLP*, volume 4, pp. 2399–2402. IEEE, 1996.

[15] O. Cetin, A. Kantor, S. King, C. Bartels, M. Magimai-Doss, J. Frankel, and K. Livescu. An articulatory feature-based tandem approach and factored observation modeling. In *Proc. ICASSP*, volume 4, pp. IV–645. IEEE, 2007.

[16] L. L. Chase. *Error-responsive feedback mechanisms for speech recognizers*. PhD thesis, Carnegie Mellon University Pittsburgh, PA, 1997.

[17] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proc. ACL*, pp. 310–318, 1996.

[18] S. Cox and S. Dasmahapatra. High-level approaches to confidence estimation in speech recognition. *IEEE Trans. on Speech and Audio Processing*, 10(7):460–471, 2002.

[19] S. Cox and R. Rose. Confidence measures for the switchboard database. In *Proc. ICASSP*, volume 1, pp. 511–514. IEEE, 1996.

[20] S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 28(4):357–366, 1980.

[21] R. E. Donovan. Segment pre-selection in decision-tree based speech synthesis systems. In *Proc. ICASSP*, 2000.

[22] R. E. Donovan and P. C. Woodland. Improvements in an HMM-based speech synthesiser. In *Eurospeech Proceedings: 4th European Conference on Speech Communication and Technology*, volume 1, pp. 573–576, 1995.

[23] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, 2012.

[24] G. Evermann and P. C. Woodland. Large vocabulary decoding and confidence estimation using word posterior probabilities. In *Proc. ICASSP*, volume 3, pp. 1655–1658. IEEE, 2000.

[25] M. Federico, N. Bertoldi, and M. Cettolo. IRSTLM: an open source toolkit for handling large scale language models. In *Proc. Interspeech*, pp. 1618–1621, 2008.

[26] M. J. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech & Language*, 12(2):75–98, 1998.

[27] M. J. Gales. Semi-tied covariance matrices for hidden Markov models. *IEEE Trans. on Speech and Audio Processing*, 7(3):272–281, 1999.

[28] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett. Darpa TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. *NASA STI/Recon Technical Report N*, 93:27403, 1993.

[29] L. Gillick, Y. Ito, and J. Young. A probabilistic approach to confidence estimation and evaluation. In *Proc. ICASSP*, volume 2, pp. 879–882. IEEE, 1997.

[30] M. Goldstein. Classification of methods used for assessment of text-to-speech systems according to the demands placed on the listener. *Speech Communication*, 16(3):225–244, 1995.

[31] R. A. Gopinath. Maximum likelihood modeling with Gaussian distributions for classification. In *Proc. ICASSP*, volume 2, pp. 661–664. IEEE, 1998.

[32] H. Harrod. How do you teach a computer to speak like Scarlett Johansson? http://goo.gl/xn5gBw, 2014. [Online; accessed 15-February-2014].

[33] T. Hirai and S. Tenpaku. Using 5 ms segments in concatenative speech synthesis. In *Fifth ISCA Workshop on Speech Synthesis*, 2004.

[34] H. Hon, A. Acero, X. Huang, J. Liu, and M. Plumpe. Automatic generation of synthesis units for trainable text-to-speech systems. In *Proc. ICASSP*, volume 1, pp. 293–296. IEEE, 1998.

[35] X. Huang, A. Acero, J. Adcock, H.-W. Hon, J. Goldsmith, J. Liu, and M. Plumpe. Whistler: A trainable text-to-speech system. In *Proc. ICSLP*, volume 4, pp. 2387–2390. IEEE, 1996.

[36] A. J. Hunt and A. W. Black. Unit selection in a concatenative speech synthesis system using a large speech database. In *Proc. ICASSP*, volume 1, pp. 373–376, 1996.

[37] H. Jiang. Confidence measures for speech recognition: A survey. *Speech Communication*, 45(4):455–470, 2005.

[38] S. O. Kamppari and T. J. Hazen. Word and phone level acoustic confidence scoring. In *Proc. ICASSP*, volume 3, pp. 1799–1802. IEEE, 2000.

[39] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne. 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):187–207, 1999.

[40] T. Kemp, T. Schaaf, et al. Estimating confidence using word lattices. In *Proc. Eurospeech*, 1997.

[41] K. Kirchhoff, G. A. Fink, and G. Sagerer. Combining acoustic and articulatory feature information for robust speech recognition. *Speech Communication*, 37(3):303–319, 2002.

[42] S. P. Kishore and A. W. Black. Unit size in unit selection speech synthesis. In *Proc. Interspeech*, 2003.

[43] D. H. Klatt. Review of text-to-speech conversion for English. *JASA*, 82(3):737–793, 1987.

[44] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *Proc. ICASSP*, volume 1, pp. 181–184. IEEE, 1995.

[45] J. Kominek. The CMU ARCTIC Speech Databases for Speech Synthesis research, 2003.

[46] J. Kominek and A. W. Black. Impact of durational outlier removal from unit selection catalogs. In *Fifth ISCA Workshop on Speech Synthesis*, 2004.

[47] R. Kumar and S. P. Kishore. Automatic pruning of unit selection speech databases for synthesis without loss of naturalness. In *Proc. Interspeech*, 2004.

[48] K.-Y. Leung and M. Siu. Phone level confidence measure using articulatory features. In *Proc. ICASSP*, volume 1, pp. I–600. IEEE, 2003.

[49] K.-Y. Leung and M. Siu. Articulatory-feature-based confidence measures. *Computer Speech & Language*, 20(4):542–562, 2006.

[50] Z. Ling and R. Wang. HMM-based unit selection using frame sized speech segments. In *Ninth International Conference on Spoken Language Processing*, 2006.

[51] H. Lu, Z. Ling, S. Wei, L. Dai, and R. Wang. Automatic error detection for unit selection speech synthesis using log likelihood ratio based SVM classifier. In *Proc. Interspeech*, volume 10, pp. 162–165, 2010.

[52] H. Lu, W. Zhang, X. Shao, Q. Zhou, W. Lei, H. Zhou, and A. Breen. Pruning Redundant Synthesis Units Based on Static and Delta Unit Appearance Frequency. In *Proc. Interspeech*, 2015.

[53] B. Maison and R. Gopinath. Robust confidence annotation and rejection for continuous speech recognition. In *Proc. ICASSP*, volume 1, pp. 389–392. IEEE, 2001.

[54] S. Matsoukas, R. Schwartz, H. Jin, and L. Nguyen. Practical implementations of speaker-adaptive training. In *DARPA Speech Recognition Workshop*. Citeseer, 1997.

[55] V. Mitra, H. Nam, and C. Y. Espy-Wilson. Robust speech recognition using articulatory gestures in a dynamic Bayesian network framework. In *Proc. ASRU*, pp. 131–136. IEEE, 2011.

[56] P. J. Moreno and C. Alberti. A factor automaton approach for the forced alignment of long speech recordings. In *Proc. ICASSP*, pp. 4869–4872. IEEE, 2009.

[57] P. Motlicek, D. Imseng, B. Potard, P. N. Garner, and I. Himawan. Exploiting foreign resources for DNN-based ASR. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(1):1–10, 2015.

[58] I. R. Murray, J. L. Arnott, N. Alm, and A. F. Newell. A communication system for the disabled with emotional synthetic speech produced by rule. In *Proc. Eurospeech*, 1991.

[59] J. P. Olive. Rule synthesis of speech from dyadic units. In *Proc. ICASSP*, volume 2, pp. 568–570. IEEE, 1977.

[60] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an ASR corpus based on public domain audio books. In *Proc. ICASSP*, pp. 5206–5210, 2015.

[61] V. Peddinti, D. Povey, and S. Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech*, 2015.

[62] J. Pinto and R. Sitaram. Confidence measures in speech recognition based on probability distribution of likelihoods. In *Proc. Interspeech*, pp. 3001–3004, 2005.

[63] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, et al. The Kaldi speech recognition toolkit. 2011.

[64] K. Prahallad and A. W. Black. Segmentation of monologues in audio books for building synthetic voices. *IEEE Transaction on Audio, Speech, and Language Processing*, 19(5):1444–1449, 2011.

[65] V. Raghavendra and K. Prahallad. Database pruning for Indian language unit selection synthesizers. pp. 67–74, 2009.

[66] B. S. R. Rajaram, K. H. R. Shiva, and R. AG. MILE TTS for Tamil for Blizzard Challenge 2014. In *Blizzard Challenge*, 2014.

[67] S. K. Rallabandi, A. Vadapalli, S. Achanta, and S. Gangashetty. IIIT Hyderabad's submission to the Blizzard Challenge 2015. In *Proc. Blizzard Challenge 2015*, 2015.

[68] B. Rueber. Obtaining confidence measures from sentence probabilities. In *Proc. Interspeech*, 1997.

[69] A. Sanchis, A. Juan, and E. Vidal. Estimating confidence measures for speech recognition verification using a˜ smoothed naive Bayes model. In *Pattern Recognition and Image Analysis*, pp. 910–918. Springer, 2003.

[70] D. Schwarz, G. Beller, B. Verbrugghe, S. Britton, et al. Real-time corpus-based concatenative synthesis with catart. In *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx), Montreal, Canada*, pp. 279–282. Citeseer, 2006.

[71] H. Segi, T. Takagi, and T. Ito. A concatenative speech synthesis method using context dependent phoneme sequences with variable length as search units. In *Proc. Fifth ISCA Workshop on Speech Synthesis*, 2004.

[72] M. Siu and H. Gish. Evaluation of word confidence for speech recognition systems. *Computer Speech & Language*, 13(4):299–319, 1999.

[73] R. Sproat, A. W. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards. Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333, 2001.

[74] P. Taylor, A. W. Black, and R. Caley. The architecture of the festival speech synthesis system. 1998.

[75] L. Tóth. Phone recognition with hierarchical convolutional deep maxout networks. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(1):1–13, 2015.

[76] W. Verhelst and M. Roelands. An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech. In *Proc. ICASSP*, volume 2, pp. 554–557, 1993.

[77] N. T. Vu, F. Kraus, and T. Schultz. Multilingual A-stabil: A new confidence score for multilingual unsupervised training. In *Proc. Spoken Language Technology Workshop*, pp. 183–188, 2010.

[78] L. Wang, Y. Zhao, M. Chu, F. K. Soong, and Z. Cao. Phonetic transcription verification with generalized posterior probability. In *Proc. Interspeech*, 2005.

[79] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, and A. Stolcke. Neural-network based measures of confidence for word recognition. In *Proc. ICASSP*, p. 887. IEEE, 1997.

[80] F. Wessel, K. Macherey, and H. Ney. A comparison of word graph and n-best list based confidence measures. In *Proc. Eurospeech*, 1999.

[81] F. Wessel, K. Macherey, and R. Schluter. Using word probabilities as confidence measures. In *Proc. ICASSP*, volume 1, pp. 225–228. IEEE, 1998.

[82] F. Wessel, R. Schlüter, K. Macherey, and H. Ney. Confidence measures for large vocabulary continuous speech recognition. *IEEE Trans. on Speech and Audio Processing*, 9(3):288–298, 2001.

[83] S. R. Young. Detecting misrecognitions and out-of-vocabulary words. In *Proc. ICASSP*, volume 2, pp. II–21. IEEE, 1994.

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

[85] T. Zeppenfeld, M. Finke, K. Ries, M. Westphal, and A. Waibel. Recognition of conversational telephone speech using the janus speech engine. In *Proc. ICASSP*, volume 3, pp. 1815–1818. IEEE, 1997.

[86] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur. Improving deep neural network acoustic models using generalized maxout networks. In *Proc. ICASSP*, pp. 215–219. IEEE, 2014.