2007

Multilingual Articulatory Features for Speech Recognition

Brian M. Ore
Wright State University

Multilingual Articulatory Features for Speech Recognition

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Engineering

by

Brian M. Ore
Department of Electrical Engineering
Wright State University

2007
Wright State University

Wright State University
SCHOOL OF GRADUATE STUDIES

April 6, 2007

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Brian M. Ore ENTITLED Multilingual Articulatory Features for Speech Recognition BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Sciencein Engineering.

Brian Rigling, Ph.D.
Thesis Director

Fred Garber, Ph.D.
Department Chair

Committee on
Final Examination

Brian Rigling , Ph.D.

Fred Garber , Ph.D.

Kefu Xue , Ph.D.

Joseph F. Thomas, Jr., Ph.D.
Dean, School of Graduate Studies

ABSTRACT

Ore, Brian . M.S., Department of Electrical Engineering, Wright State University, 2007 . *Multilingual Articulatory Features for Speech Recognition.*

Articulatory features describe the way in which the speech organs are used when producing speech sounds. Research has shown that incorporating this information into speech recognizers can lead to an improvement in system performance. The majority of previous work, however, has been limited to detecting articulatory features in a single language. In this thesis, Gaussian Mixture Models (GMMs) and Multi-Layer Perceptrons (MLPs) were used to detect articulatory features in English, German, Spanish, and Japanese. The outputs of the detectors were used to form the feature set for a Hidden Markov Model (HMM)-based phoneme recognizer. The best overall detection and recognition performance was obtained using MLPs with context. Compared to Mel-Frequency Cepstral Coefficient (MFCC)-based systems, the proposed feature sets yielded an increase of up to 4.39% correct and 5.37% accuracy when using monophone models, and an increase of up to 3.22% correct and 2.60% accuracy with triphone models. On a word recognition task, however, the MFCC systems performed better. Multilingual articulatory feature detectors were also created for all four languages using MLPs. An additional feature set was created using the multilingual detectors and evaluated on the same phoneme recognition task. Compared to the feature sets created with the language-dependent MLP detectors, the maximum decrease in system performance with monophone models was 1.44% correct and 1.72% accuracy on Japanese, and the maximum improvement in system performance with triphone models was 0.75% correct and 0.40% accuracy on Spanish. On a word recognition task, the feature sets created with the multilingual MLP detectors yielded a decrease of up to 3.75% correct and 6.01% accuracy. As a final experiment, two different procedures were investigated for combining the scores from the English GMM and MLP articulatory feature detectors. It was found that the detection performance for each articulatory feature can be improved by combining the scores from all GMM and MLP detectors.

Contents

List of Figures

List of Tables

List of Abbreviations

ASR	Automatic Speech Recognition	1
HMM	Hidden Markov Model	2
MFCC	Mel Frequency Cepstral Coefficient	2
GMM	Gaussian Mixture Model	2
MLP	Multi-Layer Perceptron	2
SVM	Support Vector Machine	2
IPA	International Phonetic Alphabet	7
FFT	Fast Fourier Transform	13
DCT	Discrete Cosine Transform	14
LM	Language Model	25
AF	Articulatory Feature	30
LDA	Linear Discriminant Analysis	31
WER	Word Error Rate	31
RASTA-PLP	Relative Spectral Transform - Perceptual Linear Prediction	31
MS	Modulation Spectrogram	31
PLP	Perceptual Linear Prediction	32
CRF	Conditional Random Field	32
KLT	Karhunen-Loéve Transform	32
DBN	Dynamic Bayesian Network	33
ANN	Artificial Neural Network	33
WSJ1-CSR	Wall Street Journal Continuous Speech Recognition corpus	34
HTK	Hidden Markov Model ToolKit	39
EM	Expectation Maximization	40
MLPwC	Multi-Layer Perceptron with Context	45
EER	Equal Error Rate	45
DET	Detection Error Tradeoff	46

Acknowledgements

First, I would like to thank Dr. Rigling for his patience and assistance throughout this entire process. I would like to thank Drs. Slyh and Anderson for all of the discussions that allowed this research to take place. Thank you also to Drs. Garber and Xue for their time and willingness to participate in my committee.

Thank you Mr. Hansen for help with preparing the speech corpora that was used in this research.

Thanks also to Mr. Hoeferlin for making sure that the computer systems were always available, and for fixing things when I broke them.

Brian M. Ore

Chapter 1

Introduction

Automatic Speech Recognition (ASR) considers the task of identifying the word sequence of a spoken utterance. Developing a system that can match a human's ability in decoding speech is desirable for a number of reasons. Speech recognizers could be used in automatic dictation systems, to give pilots voice control over aircraft functions, and to assist the hearing impaired. A speech recognition system is also a fundemental component for many higher level tasks such as speech-to-speech translation and spoken language understanding.

Developing a speech recognizer generally requires a considerable amount of effort. Commercial speech recognition systems are typically trained with hundreds of hours of speech data which can take thousands of hours to collect and transcribe. Unfortunately, the performance of these recognizers is often limited by a number of different factors, including speaking style (*i.e.*, read, spontaneous, or conversational), the number of users, and channel variability due to recording condition, type of microphone used, etc. Furthermore, successful speech recognition systems have only been developed for a fraction of the world's spoken languages.

One of the major drawbacks of current speech recognition technology is that the system performance is largely dependent on successfully modeling the structure of the language via a language model. Speech recognizers that rely soley on acoustical models to identify the phoneme sequence of an utterance typically yield poor performance. Research focused on improving the acoustical processing of speech falls into two main categories: feature extraction and modeling strategy. This

thesis focuses on extracting articulatory features (features that describe the way in which the speech organs are used to produce speech sounds) from a spoken utterance for use as the feature set for a Hidden Markov Model (HMM)-based speech recognizer.

In past research, numerous methods have been investigated for extracting articulatory information from an acoustic speech signal. Articulatory feature detectors have been developed using Gaussian Mixture Models (GMMs), Multi-Layer Perceptrons (MLPs), and Support Vector Machines (SVMs).[1] In the majority of the work performed, however, the focus has been limited to creating articulatory feature detectors for a single language. The research presented in this thesis is an extension of a previous approach to create multilingual articulatory feature detectors. Specifically, the multilingual articulatory feature detectors developed in this thesis are created using MLPs instead of GMMs, and the scores from the articulatory feature detectors are used to form a feature set, rather than combined with Mel-Frequency Cepstral Coefficients (MFCCs).

In this thesis, GMMs and MLPs are used to detect articulatory features in English, German, Spanish, and Japanese. Feature sets are created using the outputs of the detectors and evaluated on a phoneme recognition task. The best overall detection and recognition performance is obtained using MLPs that include context. Compared to MFCC-based systems, the proposed feature sets yield an increase of up to 4.39% correct and 5.37% accuracy when using context-independent models, and an increase of up to 3.22% correct and 2.60% accuracy with context-dependent models. On a word recognition task, the MFCC system outperforms the proposed feature set. MLPs are also used to create multilingual articulatory feature detectors by using speech data from all four languages. An additional feature set is created using the multilingual detectors and evaluated on the same phoneme recognition task. Compared to the feature sets created with the language-dependent MLP detectors, the change in percent correct and percent accuracy is less than 1.72% when using context-independent models, and less than 0.75% when using context-dependent models. On a word recognition task, the feature sets created with the multilingual MLP detectors yield a decrease of up to 3.75% correct and 6.01% accuracy.

An additional experiment is also performed on English to investigate whether it could be beneficial to combine the scores from the GMMs and MLPs. Two different methods of performing score

[1] Note that this list is not all-inclusive.

fusion using an MLP are considered. The best detection performance is obtained by combining the scores from all of the GMMs and MLPs in order to detect each articulatory feature. A feature set is created using the outputs of the fusion MLPs and evaluated on a phone recognition task. Compared to an MFCC-based system, the proposed feature set yields an increase of 4.26% correct and 3.78% accuracy when using context-independent models; with context-dependent models, the proposed feature set yields an increase in percent correct of 2.84% and an increaese in percent accuracy of 0.28%.

This thesis is organized as follows. The next chapter provides an overview of speech production and current ASR technology. Chapter 3 presents some of the past research done using articulatory features, provides a description of the corpora used in this thesis, describes how the articulatory feature detectors were created, and presents the detection results obtained. Chapter 4 describes how the outputs of the articulatory feature detectors were used as the feature set for a speech recognizer and presents the recognition results obtained. Chapter 5 describes the fusion experiments and presents the detection and recognition results. Chapter 6 provides a summary of the research done in this thesis and discusses possible future work. Finally, the accuracy of the individual articulatory feature detectors is given in the Appendix.

Chapter 2

Background

Speech recognition is generally approached as a statistical pattern recognition problem that can be formulated as follows. Suppose we are given a series of speech vectors $O = \{o_1, o_2, \cdots, o_T\}$, known as the observation sequence, and a list of all possible word sequences $\{W_i\}$. The task of the speech recognizer is to choose the most likely word sequence $\hat{W}$ for the given observation sequence; *i.e.* to compute the following

$$\hat{W} = \underset{i}{\operatorname{argmax}} \{P(W_i|O)\}. \tag{2.1}$$

Using Baye's Rule, we can express the probability of the i^{th} word sequence given the observation sequence as follows

$$P(W_i|O) = \frac{P(O|W_i)P(W_i)}{P(O)}. \tag{2.2}$$

Since the probability of the observation sequence $P(O)$ is the same for all word sequences to be considered, it can be ignored when computing Equation 2.2. Theoretically, for each possible word sequence W_i, we need to calculate $P(O|W_i)$ and $P(W_i)$, multiply these quantities, and choose the word sequence that gives the maximum

$$\hat{W} = \underset{i}{\operatorname{argmax}} \{P(O|W_i)P(W_i)\}. \tag{2.3}$$

For large vocabularies, direct evaluation of Equation 2.3 results in a massive search problem. In order to find a good word sequence in a reasonable amount of time, practical speech recognizers

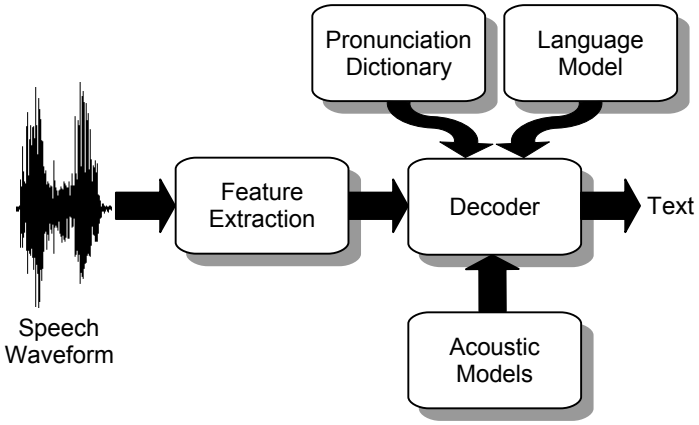


Figure 2.1: ASR flowchart.

make a number of simplifications and approximations to reduce the computational requirements.

A speech recognizer is typically implemented as follows. First, the feature vectors O are extracted fom the acoustic waveform. Next, these speech vectors are passed on to a decoder that computes Equation 2.3 and outputs the hypothesized word sequence. Acoustic models are used to calculate $P(O|W_i)$ and a language model is used to estimate the probability of the word sequence $P(W_i)$. A pronunciation dictionary specifies the sequence of phonemes used to pronounce each word. Figure 2.1 illustrates this process.

The following section gives a brief overview of the speech production process and discusses how different speech sounds are commonly described. The remaining sections of this chapter discuss each process shown in Figure 2.1—namely, feature extraction, acoustic modeling, language modeling, and decoding. Section 2.2 descibes how Mel-Frequency Cepstral Coefficients (MFCCs) are extracted from the acoustic speech signal. MFCCs are currently one of the most popular feature sets used in state-of-the-art speech recognizers. Section 2.3 presents an overview of Hidden Markov Model (HMM) theory, which is a pre-requistite for discussing acoustic modeling and decoding. Acoustic modeling is discussed in Section 2.4 and language modeling is introduced in Section 2.5. Finally, Section 2.6 describes the decoder.

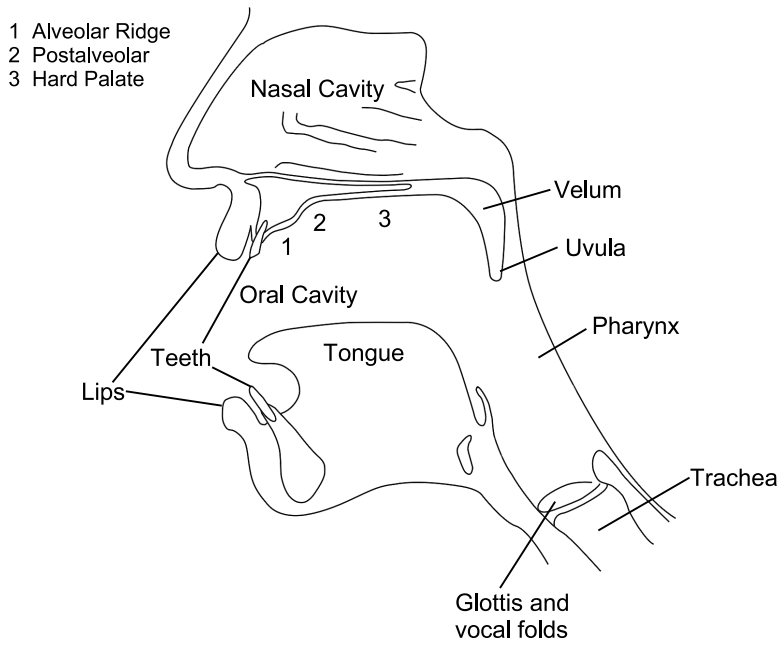


Figure 2.2: Primary speech organs; adapted from [1].

2.1 Speech Production

This section provides an overview of the speech production process [1]. The main parts of the human body used to produce speech sounds are shown in Figure 2.2. Together with the lungs, these speech organs form the vocal tract. The vocal tract is normally divided into two sections: the section below the glottis, known as the subglottal vocal tract, and the section above the glottis, known as the supraglottal vocal tract. In general, the speech organs of the subglottal vocal tract provide the source of energy for speech production, whereas the supraglottal vocal tract determines the speech quality.

The majority of all speech sounds are produced using an outward flow of air from the lungs as the energy source. This flow of air travels up from the lungs through the trachea until it reaches the glottis, which is a thin aperture formed by the the vocal folds. The airstream from the glottis can be varied in several ways. When the vocal folds are tensed, the glottis is closed and air is prevented from escaping the lungs. Due to respiratory forces, air pressure from the lungs builds up until it momentarily forces the folds apart; as the air is released the pressure is reduced and the glottis closes again. This process results in air being released from the lungs in a quasi-periodic manner, and this vibration of the vocal folds is known as phonation. Speech produced in this manner is

commonly referred to as voiced speech. If the vocal folds are held far enough apart, and the airflow from the lungs is moderate, there will be an absence of phonation and the resulting speech will be unvoiced.

Once the airstream from the lungs has passed through the glottis it enters the pharynx, which extends from the glottis up to the base of the skull. The pharynx can be varied in geometry by tongue movement and depending on the state of the velum, leads into both the oral and nasal cavities. When the velum is in its relaxed position (as shown in Figure 2.2), air is allowed to flow into both the oral and nasal cavity; when the velum is raised, the nasal cavity is blocked off so that the airflow is directed into the oral cavity. Airflow directed into the nasal cavity passes out the nose and airflow directed into the oral cavity exits through the mouth. The shape of the oral cavity can be varied in a number of ways to produce different speech sounds: the tongue position, extent to which the jaw is opened, and the shape of the lips all contribute to changing the geometry of the oral cavity.

The two major classes that speech sounds are categorized into are vowels and consonants. Vowels are produced by varying the shape of the pharyngeal and oral cavities such that the airflow from the glottis is relatively unobstructed. Consonants are generally formed by either constricting or blocking the airstream by using the tongue, teeth, and lips. One approach that has been devised to universally represent these speech sounds across different languages is the International Phonetic Alphabet (IPA) [2]. The next two sections discuss the ways in which vowels and consonants are commonly produced and categorized, using the IPA scheme.

2.1.1 Vowels

Vowels are produced by varying the shape of the supraglottal vocal tract such that the airflow from the glottis is not constricted. All vowels are normally voiced and the majority of vowel sounds are produced with the velum raised so that the airflow from the pharynx is directed into the oral cavity. The geometry of the pharyngeal and oral cavity is controlled primarily by the tongue and the lips. In order to describe the shape and position of the tongue, as well as the shape of the lips, vowels are normally plotted on a vowel chart. Figure 2.3 shows the IPA vowel chart, which describes the shape and position of the tongue for a particular vowel in terms of tongue height (vertical axis) and tongue fronting (horizontal axis). The boundaries of the vowel chart represent contraints such that

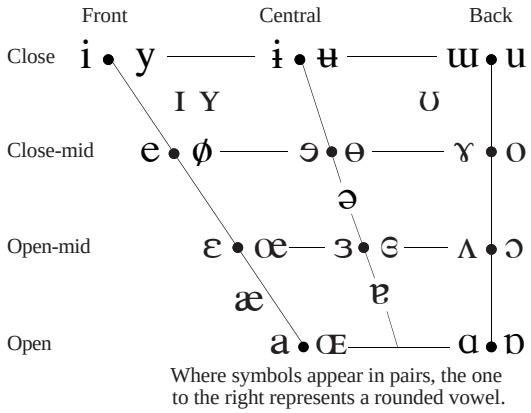


Figure 2.3: IPA vowel chart.

if the tongue moves beyond these limits, it will create a contriction in the vocal tract and the sound produced will no longer be vocalic. The divisions along the vertical and horizontal axis represent vowel sounds that are judged to be equidistant from one another in terms of tongue height and tongue fronting, respectively.

The shape of the lips when producing vocalic sounds can be described as being rounded or unrounded. The IPA vowel chart represents lip position as follows: where vowels appear in pairs, such as /i/ and /y/ in Figure 2.3, the vowel on the left is unrounded and the vowel on the right is rounded; vowels that do not appear in pairs can be either rounded or unrounded.

The vowel sounds discussed so far all have a target articulatory configuration; that is, the tongue and lips have a particular shape and position that they form. Some vowels, however, are produced in a manner such that there is no single target configuration that they are meant to achieve. These vowels are referred to as diphthongs and are represented by concatenating the symbols for the start and end configuration; for example, a vowel sound with the start configuration /a/ and end configuration /ɪ/ is written as /aɪ/.

2.1.2 Consonants

Consonants are produced by either constricting or blocking the airstream from the lungs using the tongue, teeth, and lips. Consontants can can be either voiced or unvoiced, and are commonly described by the place and manner of articulation. Place of articulation refers to the location in

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Where symbols appear in pairs, the one to the right represents a voiced consonant. Shaded areas denote articulations judged impossible.

Figure 2.4: IPA consonant chart.

Table 2.1: Places of articulation.

Place	Articulators
Bilabial	Upper and lower lips
Labiodental	Lower lip and upper teeth
Dental	Upper teeth
Alveolar	Upper surface of the mouth immediately behind the front teeth
Postalveolar	Extends from the alveolar ridge to the start of the hard palate
Retroflex	Tip of the tongue and the back part of the alveolar ridge
Palatal	Hard palate
Velar	Soft palate, or velum
Uvular	Uvula
Pharyngeal	Pharynx walls
Glottal	Glottis

the vocal tract where the constriction is made; manner of articulation refers to the degree to which the airstream is constricted and, in some cases, the shape of the constriction. Figure 2.4 shows the IPA consonant chart. A brief description of the places of articulation recognized by the IPA scheme is given in Table 2.1. The following list introduces the eight manners of articulation that are acknowledged by the IPA—namely, plosives, nasals, trills, taps or flaps, fricatives, lateral fricatives, approximants, and lateral approximants.

- Plosives are produced by momentarily blocking the airstream from the lungs at some point along the supraglottal vocal tract and then releasing the air; the release of this closure causes a burst of turbulent airflow to be produced.

- Nasals are formed by completely blocking the airflow through the oral cavity while simultaneously lowering the velum.

- Trills are produced by holding one articulator loosely near another so that a flow of air between the two articulators causes one of them to vibrate.

- Taps and flaps are characterized as very brief obstructions in the supraglottal vocal tract. A tap is produced by deliberately moving an articulator, typically the tongue tip, to create the closure. Flaps are produced in a similar manner except that the contact is an unintentional effect of producing the speech sound.

- Fricatives are produced by partially obstructing the airstream from the lungs by narrowing the distance between two articulators so that turbulent airflow is created.

- Lateral fricatives are fricatives produced in a manner such that the airflow from the lungs is directed around the sides of the tongue.

- Approximants are formed by holding one articulator close to another so that a constriction is created, but turblent airflow is not produced.

- Lateral approximants are approximants produced in a manner such that the airflow from the lungs is directed around the sides of the tongue.

2.2 Mel-Frequency Cepstral Coefficients

Mel-Frequency Cepstral Coefficients (MFCCs) are one of the most popular feature sets used in state-of-the-art speech recognizers. This section provides an overview of how MFCCs can be extracted from an acoustic speech waveform. Section 2.2.1 introduces the Mel scale. Section 2.2.2 describes several preprocessing operations that are usually performed, and Section 2.2.3 discusses how MFCCs are obtained from the preprocessed speech signal. Finally, Section 2.2.4 describes several coefficients that are commonly included in addition to MFCCs.

2.2.1 Mel Scale

Research has shown that human perception of the frequency content of sounds does not directly correspond to the values that are derived from measurements. The relationship between pitch, or

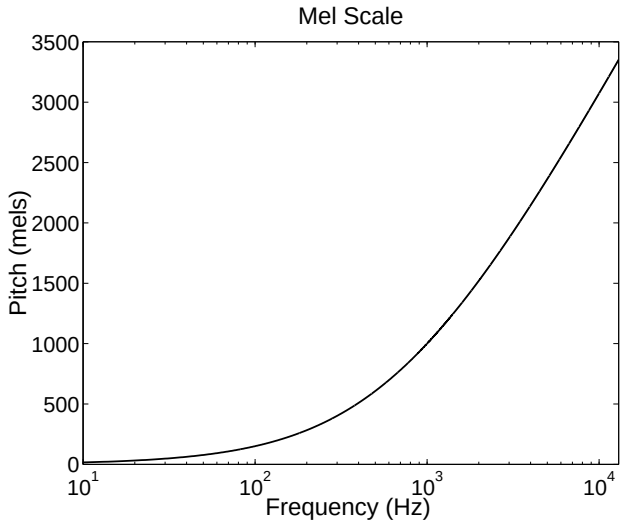


Figure 2.5: Mel scale in logarithmic frequency scaling; the Mel scale is approximately linear below 1000Hz and logarithmic above.

the perceived frequency of a sound wave, and the actual frequency is commonly described using a Mel scale. The Mel scale is a scale of pitches that are equidistant from one another and was first introduced in an experiment by Stevens and Volkmann [3]. As a reference point, the pitch of a 1000 Hz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 mels. Participants in Stevens and Volkmann's experiment were asked to adjust the frequency of the reference tone until the pitch they perceived was $\frac{1}{2}$ the reference, twice the reference, and so on. These pitches were labeled as 500 mels, 2000 mels, etc. The Mel scale is commonly approximated as

$$\text{Mel}(f) = 2595 \log 10 \left(1 + \frac{f}{700}\right). \tag{2.4}$$

Figure 2.5 shows the Mel scale as a function of logarithmic frequency scaling. This scale is approximately linear below 1000 Hz and logarithmic above.

2.2.2 Preprocessing

The statistical properties of speech vary over time; the transforms used for processing speech signals, however, assume a stationary signal. The common solution to this problem is to block the original sampled waveform into overlapping frames that are short enough so that we can consider each frame of speech as stationary. If we denote the original sampled speech waveform as $d(n)$,

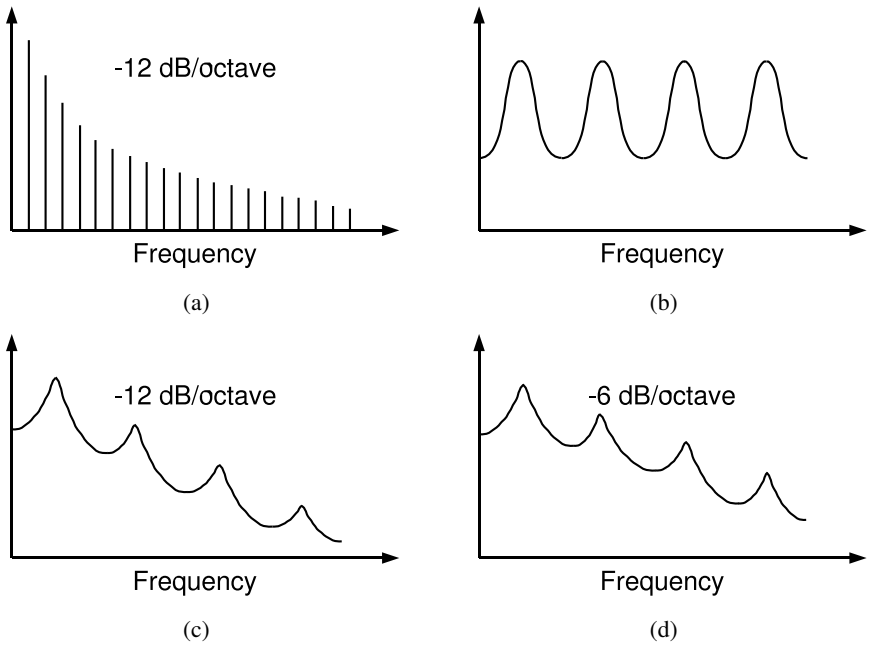


Figure 2.6: Spectral properties of the vowel /ɜ/ (a) spectrum of the phonation source (b) vocal tract frequency response (c) spectral envelope resulting from filtering the phonation source with the vocal tract (d) spectral envelope of the emitted acoustic wave; adapted from [1].

then the l^{th} frame of speech $s_l(n)$ can be formed as

$$s_l(n) = d(Ml + n), \qquad 0 \leq n < N; \quad 0 \leq l < L, \tag{2.5}$$

where M is the number of samples between consecutive frame beginnings, N is the number of samples in each frame, and L is the total number of frames. The values M and N are typically chosen so that each frame of speech is approximately 20-40ms in duration and the overlap is 50-75%. Once the frame blocking procedure is completed, the DC mean is removed from each frame. This is useful for removing any DC offset that may have been introduced in the original analog-to-digital conversion [9].

Voiced speech output from the lips is influenced by several factors, including the acoustic properties of phonation, the vocal tract, and the acoustic properties of the human head. Figure 2.6 shows these components for the vowel /ɜ/, specifically (a) the spectrum of the phonation source (b) the frequency response of the vocal tract (c) the spectral envelope resulting from filtering the phonation source with the vocal tract, and (d) the spectral envelope of the final acoustic wave. In

order to flatten the spectrum of the speech signal output from the lips, a pre-emphasis filter can be applied to add a boost of $+6$ dB per octave. The frequency response of the pre-emphasis filter can be expressed as

$$H(z) = 1 - uz^{-1}, \tag{2.6}$$

where $u = 0.97$. The pre-emphasis operation is applied to the l^{th} frame as

$$s'_l(n) = s_l(n) - us_l(n-1), \qquad 0 \leq n < N. \tag{2.7}$$

where $u = 0.97$. To attenuate the discontinuities at the edges of each frame, a windowing function is typically applied. One of the most popular windowing functions is the Hamming window, which is applied to the l^{th} frame of speech as

$$s'_l(n) = \left\{0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right)\right\} s_l(n), \qquad 0 \leq n < N. \tag{2.8}$$

2.2.3 Filterbank and Cepstral Analysis

To obtain the non-linear frequency resolution of the Mel scale, filterbank analysis is commonly performed. This is accomplished by using a series of B triangular filters spaced along the frequency axis to give approximately equal resolution on the Mel scale. As an example, suppose we want to design a filterbank that covers the frequency range $[0, 8000\text{Hz}]$. The center frequency P_c (Mels) of each filter can be calculated as follows

$$P_c = \frac{\text{Mel}(8000)}{(B+1)}\, c, \qquad 1 \leq c \leq B. \tag{2.9}$$

The corresponding frequency in Hertz f_c can be obtained by solving Equation 2.4 for f. Next, a series of triangular filters with center frequencies $\{f_1, f_2, \cdots, f_B\}$ can be designed such that the lower and upper cut-offs are the neighboring center frequencies. Figure 2.7 shows a filterbank with $B = 20$ triangular filters.

The filterbank analysis is performed by first calculating the FFT of the windowed speech signal. Next, the magnitude of each coefficient is multiplied by the corresponding filter gains and the results

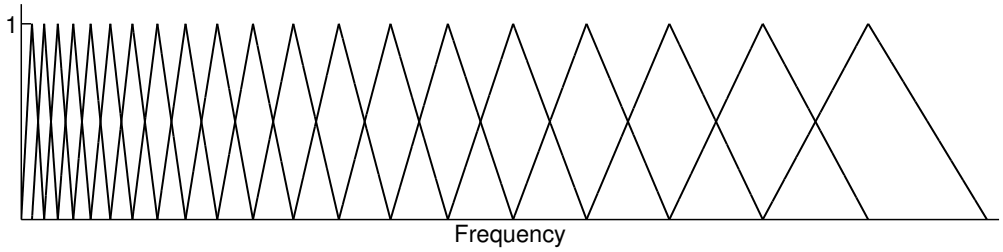


Figure 2.7: Filterbank with $B = 20$ triangular filters.

for each of the triangular filters are accumulated. Denote the sum of each filter for the l^{th} frame of speech as $m_l(j)$, where $j = 1, 2, \ldots, B$.

MFCCs are calculated from the log filterbank sums using the Discrete Cosine Transform (DCT). The DCT is used to decorrelate the coefficients so that diagonal covariance matrices can be used in the HMM system. Typically $I = 12$ coefficients are calculated for the l^{th} frame of speech as follows

$$c_l(i) = \sqrt{\frac{2}{B}} \sum_{j=1}^{B} \log_{10}\{m_l(j)\} \cos\left(\frac{\pi i}{B}(j - 0.5)\right), \qquad 1 \leq i \leq I. \tag{2.10}$$

Next, the coefficients are liftered so that they have a similar range of values. A sinusoidal lifter is applied using the following formula

$$c_l'(i) = \left(1 + \frac{D}{2} \sin\frac{\pi i}{D}\right) c_l(i), \tag{2.11}$$

where $D = 22$. A final operation that is commonly performed is cepstral mean subtraction, which is the removal of the mean value of each coefficient. This is useful for compensating for long term spectral effects caused by different microphones and audio channels [4].

2.2.4 Energy and Derivative Coefficients

In addition to MFCC features, the log of the signal energy is often calculated. One problem that is encountered with this feature is that it varies greatly depending on recording condition and speaker. In order to reduce this variance, the energy feature can be normalized to the range $[E_{min}, 1, 0]$, where E_{min} is the minimum log signal energy calculated over all L frames.

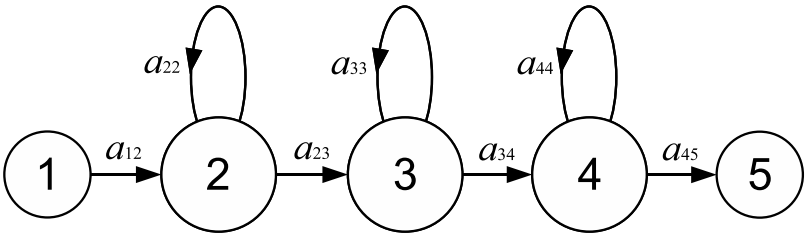


Figure 2.8: Five state HMM used to model phonemes; states one and five are non-emitting.

The performance of a speech recognition system can be enhanced by estimating time derivatives of all features [5]. Both first order (delta) and second order (acceleration) coefficients are commonly used. The delta MFCCs for the l^{th} frame of speech are calculated as follows

$$dc_l(i) = \frac{\sum_{\theta=1}^{2} \theta \left(c_{l+\theta}(i) - c_{l-\theta}(i) \right)}{2 \sum_{\theta=1}^{2} \theta^2}, \qquad 1 \leq i \leq I. \tag{2.12}$$

The second order coefficients are calculated by the same formula, using the first order coefficients as input. Time derivatives of the energy features are calculated in a similar manner.

2.3 Hidden Markov Models

A Hidden Markov Model (HMM) is a statistical model commonly used to represent time series data. An HMM can be used to model either continuous or discrete valued data; although, in this section we will assume that the data is continuous valued. HMMs are typically defined in terms of the elements that constitute the model—namely, the number of states in the model, the state transition probability distributions, and the observation probability density functions for each emitting state.[1] Figure 2.8 shows the general structure of an HMM used to model phonemes.

Suppose we describe speech production as a process that can be decomposed into N stages. For example, most phonemes are roughly produced in three stages: when the articulators (1) move toward the target configuration, (2) are in the target configuration, or (3) move away from the target

[1] HMM states can be either emitting or non-emitting; that is, the state produces observations (emitting) or the state does not produce observations (non-emitting).

configuration (toward the next target). In the HMM system framework, each of these stages is represented as an emitting state; denote the state at time t as $x(t)$. At each discrete time t, the HMM system produces an observation o_t (provided that the current state is emitting) and either remains in the same state or transitions to a new state. The state transition probability distribution $A = \{a_{ij}\}$ defines the probability of transitioning from state i to state j, where

$$a_{ij} = P(x(t) = j | x(t-1) = i), \qquad 1 \leq i, j \leq N. \tag{2.13}$$

The observation probability density functions $b_j(o)$ are used to calculate the probability that the observation o_t was produced by state j. The most common type of observation probability density functions used are Gaussian mixture densities, which are of the form

$$b_j(o) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(o; \mu_{jm}, \Sigma_{jm}), \tag{2.14}$$

where M is the number of mixture components, c_{jm} is the weight of the m^{th} mixture component for state j, and $\mathcal{N}(o; \mu, \Sigma)$ is a single Gaussian density defined as

$$\mathcal{N}(o; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left\{ -\frac{1}{2}(o-\mu)^T \Sigma^{-1} (o-\mu) \right\} \tag{2.15}$$

with mean vector μ, covariance matrix Σ, and n is the dimensionality of o. Denote the complete model as follows

$$\lambda = \{(A, c_{jm}, \mu_{jm}, \Sigma_{jm}) | j = 2, 3, \cdots, N-1, m = 1, 2, \cdots, M\}. \tag{2.16}$$

Consider the model shown in Figure 2.8. Note that states one and five are non-emitting; that is, they do not produce observations. In addition, let us constrain the model such that the initial state is always state one and the final state is always state five. In the following sections we will assume that the HMMs we are considering are of a similar form; that is, the initial and final states are non-emitting. The following sections introduce two basic assumptions used in HMM theory and consider three fundamental problems that arise when dealing with HMMs [6, 7], namely

1. **Evaluating HMMs:** Given an observation sequence $O = \{o_1, o_2, \cdots, o_T\}$ and a model λ,

how do we calculate the probability that the observation sequence was produced by the model $P(O|\lambda)$?

2. **Decoding HMMs:** For a given observation sequence $O = \{o_1, o_2, \cdots, o_T\}$ and model λ, what is the single best state sequence $\hat{X} = \{\hat{x}(0), \hat{x}(1), \cdots \hat{x}(T+1)\}$; that is, the state sequence that maximizes $P(X|O, \lambda)$?

3. **Training HMMs:** Given an observation sequence $O = \{o_1, o_2, \cdots, o_T\}$ and a model λ, how do we update the model parameters such that the probability of the given observation is maximized?

2.3.1 HMM Assumptions

Two important assumptions used in HMM theory are the Markov Assumption and the Output-Independence Assumption [8]. The Markov Assumption states that the the probability of being in a particular state at a given time depends only on the state at the previous time

$$P(x(t)|x(0), x(1), \cdots, x(t-1)) = P(x(t)|x(t-1)). \tag{2.17}$$

The Output-Independence Assumption states that the probability of an observation at a particular time depends only on the current state and is independent of the past observations

$$P(o_t|o_{t-1}, o_{t-2}, \cdots, o_1, x(t-1), x(t-2), \cdots, x(0)) = P(o_t|x(t)). \tag{2.18}$$

2.3.2 Evaluating HMMs

The most straightforward method of obtaining $P(O|\lambda)$ is to consider all possible state sequences of length $T+2$ and to sum the probabilities that the given observation sequence was generated.[2] Consider first one such possible state sequence

$$X = x(0), x(1), \cdots, x(T+1), \tag{2.19}$$

[2]Note that for an observation sequence of length T the state sequence is length $T+2$ because the initial and final states are non-emitting.

where $x(0)$ is the initial state and $x(T+1)$ is the final state. To calculate the probability of the observation sequence O for the state sequence of 2.19, we can apply the Output-Independence Assumption 2.18 to obtain

$$P(O|X,\lambda) = \prod_{t=1}^{T} P(o_t|x(t),\lambda) = b_{x(1)}(o_1)b_{x(2)}(o_2)\cdots b_{x(T)}(o_T). \tag{2.20}$$

By applying the Markov Assumption 2.17, the probability of the state sequence Q can be written as follows

$$P(X|\lambda) = \prod_{t=1}^{T+1} P(x(t)|x(t-1),\lambda) = a_{x(0)x(1)}a_{x(1)x(2)}a_{x(2)x(3)}\cdots a_{x(T)x(T+1)}. \tag{2.21}$$

The probability of the observation sequence O and the state sequence Q occurring simultaneously, or the joint probability of O and Q, is obtained by using the product rule

$$\begin{aligned} P(O,X|\lambda) &= P(O|X,\lambda)P(X|\lambda) \\ &= a_{x(0)x(1)}b_{x(1)}(o_1)a_{x(1)x(2)}b_{x(2)}(o_2)\cdots b_{x(T)}(o_T)a_{x(T)x(T+1)}. \end{aligned} \tag{2.22}$$

Thus, Equation 2.22 gives us an expression for the probability of the observation sequence and one possible state sequence. We can now obtain $P(O|\lambda)$ by summing over all possible state sequences

$$\begin{aligned} P(O|\lambda) &= \sum_{\text{all } X} P(O|X,\lambda)P(X|\lambda) \\ &= \sum_{\text{all } X} a_{x(0)x(1)}b_{x(1)}(o_1)a_{x(1)x(2)}b_{x(2)}(o_2)\cdots b_{x(T)}(o_T)a_{x(T)x(T+1)}. \end{aligned} \tag{2.23}$$

2.3.3 Decoding HMMs

In the previous section, we found that we can compute the probability that an observation sequence was produced by a model, *i.e.* $P(O|\lambda)$, by summing over all possible state sequences. This section considers the problem of finding the single best state sequence $\hat{X} = \{\hat{x}(0), \hat{x}(1), \cdots \hat{x}(T+1)\}$ for a given observation sequence and model—that is, the state sequence that maximizes $P(X|O,\lambda)$.

Algorithm 1 Viterbi Algorithm

Step 1: Initialization	
$\phi_j(1) = a_{1j} b_j(o_1)$	$1 < j < N$
$\phi_1(1) = 1$	
Step 2: Recursion	
$\phi_j(t) = \max\limits_{1<i<N} \{\phi_i(t-1) a_{ij}\} b_j(o_t)$	$2 \le t \le T; \quad 1 < j < N$
$\psi_j(t) = \operatorname{argmax}\limits_{1<i<N} \{\phi_i(t-1) a_{ij}\}$	
Step 3: Termination	
$\hat{P}(O\|\lambda) = \max\limits_{1<i<N} \{\phi_i(T) a_{iN}\}$	
$\hat{x}(T) = \operatorname{argmax}\limits_{1<i<N} \{\phi_i(T) a_{iN}\}$	
Step 4: Backtracking	
$\hat{x}(t) = \psi_{\hat{x}(t+1)}(t+1)$	$t = T-1, T-2, \cdots, 1$

Consider first the following equality

$$P(X|O,\lambda) = \frac{P(X,O|\lambda)}{P(O|\lambda)}. \tag{2.24}$$

Since $P(O|\lambda)$ is a constant for all state sequences X under consideration, we can equivalently find a state sequence that maximizes $P(X,O|\lambda)$. One method that has been developed for finding such a state sequence is the Viterbi Algorithm, which is described in Algorithm 1 [8, 9]. Define the quantity $\phi_j(t)$ as the probability of the most likely state sequence up to time t that accounts for the first t observations and ends in state j, that is

$$\phi_j(t) = \max_{x(0),x(1),\cdots,x(t-1)} P(x(0), x(1), \cdots, x(t) = j, o_1, o_2, \cdots, o_t|\lambda). \tag{2.25}$$

The Viterbi Algorithm recursively computes $\phi_j(t)$ for each emitting HMM state j at time $t = 1, 2, \cdots, T$ and chooses the state sequence that maximizes $P(X,O|\lambda)$. This algorithm can be illustrated in a lattice structure, as shown in Figure 2.9. The vertical axis is the HMM state and the horizontal axis is time. Each node in the lattice represents the probability that state j generated the observation o_t, and each arc represents the probability of transitioning from one state to the next.

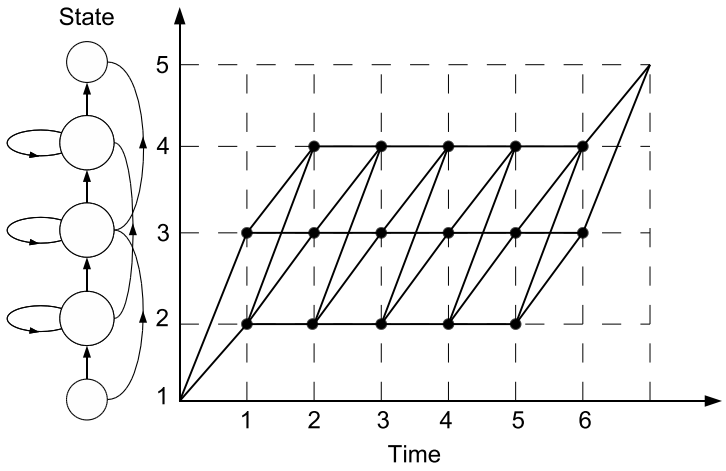


Figure 2.9: Viterbi Algorithm shown in a lattice structure; nodes represent the probability that a state generated an observation and each arc represents the probability of transitioning from one state to the next; adapted from [9].

2.3.4 Training HMMs

Given an observation sequence $O = \{o_1, o_2, \cdots, o_T\}$ and a model λ, how do we update the model parameters such that the probability of the given observation is maximized? Although there is no known way to analytically solve for the model parameters, iterative procedures exist for locally maximizing $P(O|\lambda)$. One of these methods, known as the Baum-Welch Algorithm [10], is introduced in this section. Let us begin by defining the following quantities

$$\begin{aligned}
\alpha_i(t) &= P(o_1, o_2, \cdots, o_t, x(t) = i|\lambda) \\
\beta_i(t) &= P(o_{t+1}, o_{t+2}, \cdots, o_T|x(t) = i, \lambda) \\
L_i(t) &= P(x(t) = i|O, \lambda) \\
\xi_{ij}(t) &= P(x(t) = i, x(t+1) = j|O, \lambda).
\end{aligned}$$

The first quantity $\alpha_i(t)$, also known as the forward variable, is defined as the probability of the partial observation sequence $\{o_1, o_2, \cdots, o_t\}$ and state i at time t, given the model λ. The forward variable can be calculated in a recursive manner as follows

$$\alpha_i(t) = \left[\sum_{j=2}^{N-1} \alpha_j(t-1)a_{ji}\right] b_i(o_t) \tag{2.26}$$

with the initial conditions

$$\alpha_1(1) = 1 \tag{2.27}$$

$$\alpha_i(1) = a_{1i}b_i(o_1) \tag{2.28}$$

and final condition

$$\alpha_N(T) = \sum_{i=2}^{N-1} \alpha_i(T)a_{iN}. \tag{2.29}$$

The next quantity $\beta_t(i)$ is referred to as the backward variable and can be defined as the probability of the partial observation sequence $\{o_{t+1}, o_{t+2}, \cdots, o_T\}$ given state s_i at time t and the model λ. This variable can also be calculated in a recursive manner, although whereas the recursion for the forward variable moves forward in time, the recursion for the backward variable moves backward in time. In terms of the HMM parameters, the backward variable can be calculated as follows

$$\beta_i(t) = \sum_{j=2}^{N-1} a_{ij}b_j(o_{t+1})\beta_j(t+1) \tag{2.30}$$

with the initial condition

$$\beta_i(T) = a_{iN} \tag{2.31}$$

and final condition

$$\beta_1(1) = \sum_{j=2}^{N-1} a_{1j}b_j(o_1)\beta_j(1). \tag{2.32}$$

The term $L_i(t)$ is defined as the probability of state i at time t given the observation sequence O and the model λ. To calculate this quantity, note that multiplying the forward and backward variables for state i at time t yields the probability of the observation sequence O and state i at time t, that is

$$\begin{aligned} \alpha_i(t)\beta_i(t) &= P(o_1, o_2, \cdots, o_t, x(t) = i|\lambda)P(o_{t+1}, o_{t+2}, \cdots, o_T|x(t) = i, \lambda) \\ &= P(O, x(t) = i|\lambda). \end{aligned} \tag{2.33}$$

We can now write $L_i(t)$ in terms of the forward and backward variables as follows

$$L_i(t) = \frac{P(O, x(t) = i|\lambda)}{P(O|\lambda)} = \frac{\alpha_i(t)\beta_i(t)}{P(O|\lambda)}. \tag{2.34}$$

Note that summing $L_i(t)$ over t yields the expected number of times in state i, or equivalently, the expected number of transitions from state i

$$\sum_{t=1}^{T} L_i(t) = \text{expected number of transitions from state } i. \tag{2.35}$$

Lastly, define the quantity $\xi_{ij}(t)$ as the probability of state i at time t and state j at time $t+1$, given the observation sequence O and the model λ. Note here that the probability of the observation o_{t+1} and state j at time $t+1$, given state i at time t can be written as follows

$$P(o_{t+1}, x(t+1) = j | x(t) = i, \lambda) = a_{ij} b_j(o_{t+1}). \tag{2.36}$$

Using Equation 2.36, $\xi_{ij}(t)$ can be expressed in terms of the forward and backward variables as follows

$$\xi_{ij}(t) = \frac{P(x(t) = i, x(t+1) = j, O|\lambda)}{P(O|\lambda)} = \frac{\alpha_i(t) a_{ij} b_j(o_{t+1}) \beta_j(t+1)}{P(O|\lambda)}. \tag{2.37}$$

Note that summing $\xi_{ij}(t)$ over t yields the expected number of transitions from state i to state j

$$\sum_{t=1}^{T-1} \xi_{ij}(t) = \text{expected number of transitions from state } i \text{ to state } j. \tag{2.38}$$

Given the quantities $L_i(t)$ and $\xi_{ij}(t)$, the transition probabilities can be estimated as follows. The probability of transitioning from the initial non-emitting state 1 to state j is simply the probability of state j at time $t = 1$

$$\hat{a}_{1j} = L_j(1), \qquad 1 < j < N. \tag{2.39}$$

The probability of transitioning from state s_i to state s_j, where s_i and s_j are emitting HMM states, can be estimated as follows

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T} L_i(t)}, \qquad 1 < i, j < N. \tag{2.40}$$

i.e., the expected number of transitions from state i to j divided by the expected number of transitions from state i. Lastly, the probability of transitioning from state i to the final non-emitting state

N is the probability of state i at time $t = T$ divided by the expected number of transitions from state i

$$\hat{a}_{iN} = \frac{L_i(T)}{\sum_{t=1}^{T} L_i(t)}, \qquad 1 < i < N. \tag{2.41}$$

In order to estimate the remaining model parameters, define the quantity $L_{jm}(t)$ as the probability that the m^{th} mixture component for state j generated the observation o_t

$$L_{jm}(t) = P(x(t) = j, U_{tj} = m|O, \lambda) = L_j(t)\frac{c_{jm}b_{jm}(o_t)}{b_j(o_t)}, \tag{2.42}$$

where U_{tj} is a random variable indicating the mixture component for state j at time t. The mixture weights, means, and covariances are updated by assigning each observation vector to every state in proportion to the probability of the model being in that state when the vector was observed [11], that is

$$\begin{aligned}
\hat{c}_{jm} &= \frac{\sum_{t=1}^{T} L_{jm}(t)}{\sum_{t=1}^{T} L_j(t)} \\
\hat{\mu}_{jm} &= \frac{\sum_{t=1}^{T} L_{jm}(t) o_t}{\sum_{t=1}^{T} L_{jm}(t)} \\
\hat{\Sigma}_{jm} &= \frac{\sum_{t=1}^{T} L_{jm}(t) \left(o_t - \hat{u}_{jm}\right)\left(o_t - \hat{\mu}_{jm}\right)^T}{\sum_{t=1}^{T} L_{jm}(t)}.
\end{aligned} \tag{2.43}$$

2.4 Acoustic Modeling

Acoustic models are used to calculate the probability of an observation sequence for a given word sequence, *i.e.* $P(O|W)$. In order to accomplish this, an acoustic model for the word sequence W must be supplied. This section considers the problem of how to define and train these models.

Consider first the problem of defining the model set. One possibility is to create a model for every word in the given vocabulary and concatenate these models together to represent the entire

	the		sun			shined			
CI	ð	ə	s	ʌ	n	ʃ	aɪ	n	d
CD	ð+ə	ð-ə+s	ə-s+ʌ	s-ʌ+n	ʌ-n+ʃ	n-ʃ+aɪ	ʃ-aɪ+n	aɪ-n+d	n-d

Figure 2.10: Context-Independent (CI) and Context-Dependent (CD) modeling strategies for the word sequence *the sun shined*.

word sequence. Even with moderately sized vocabularies, however, it is commonly the case where a given word appears infrequently, or not at all, in the training corpus. A common solution to this problem is to model words at the phoneme level. Two diferent strategies that have been developed for defining such a model set are context-independent and context-dependent modeling. Context-independent modeling defines a single model for each phoneme in a given language; thus the total number of models needed to represent every word in a given language is equal to the number of phonemes in that language. Context-dependent modeling defines a single model for each phoneme in a particular context. This context is usually based on the previous and following phoneme, in which case the resulting models are referred to as triphone models. Figure 2.10 illustrates this difference.

Phonemes are commonly modeled using HMMs with a form similar to the one shown in Figure 2.8. Given a sequence of speech vectors and phonemes for an utterance, a composite HMM can be formed by concatenating the correponding phoneme models. For example, the word sequence shown in Figure 2.10 could be modeled by concatenating nine HMMs—if each phoneme is modeled using an HMM with three emitting states the composite HMM will include 27 emitting states. This model can then be trained using the Baum-Welch algorithm, as discussed in Section 2.3.4. The next section discusses state clustering, which is often neccessary when context-dependent models are used.

2.4.1 State Clustering

State clustering is used to tie the HMM states of different phoneme models. This is particularly useful when using triphone models because there is unlikely to be a substantial amount of training data available for all possible phoneme contexts. The two main strategies used to determine which

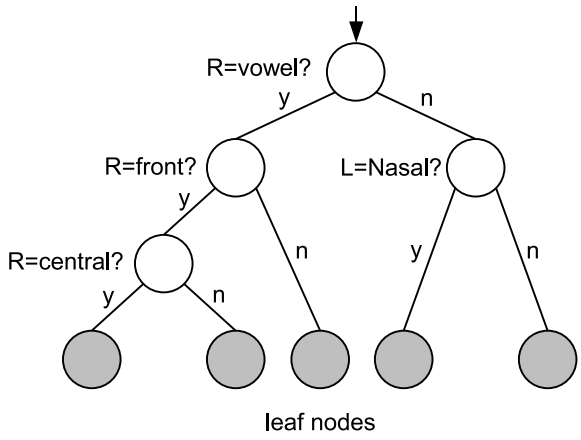


Figure 2.11: Binary decision tree where each question refers to the phoneme on the left (L) and right (R); all gray nodes are referred to as leaf nodes.

HMM states to cluster together are data-driven and decision-tree-based approaches. In this section we will only consider decision-tree-based state clustering [12].

An example of a binary decision tree is shown in Figure 2.11. In a decision tree, each node is associated with a question and the path through the tree is determined by the answers to the questions. A decision-tree for HMM state clustering can be constructed as follows. Initially, all states are placed into a single group at the root node; that is, they share a common mean and variance. Next, a question is chosen to maximize the likelihood of the training data, given the initial pool of states is split into two groups. This process continues for each node until the increase in likelihood obtained by splitting the state pool falls below some threshold. Lastly, the states in each of the leaf nodes are tied to form a single cluster.

2.5 Language Modeling

A Language Model (LM) is a statistical model that is used to estimate the probability of a word sequence. The basic procedure for training a LM is relatively straightforward and can be summarized as follows. Consider the following word sequence

$$W = \{w_1, w_2, \cdots, w_M\}, \tag{2.44}$$

where M is the number of words. By applying the chain rule, the probability of the word sequence W can be written as follows

$$\begin{aligned} P(W) &= P(w_1)P(w_2|w_1)P(w_3|w_1,w_2)\cdots P(w_M|w_1,\cdots,w_{M-1}) \\ &= \prod_{i=1}^{M} P(w_i|w_1,\cdots,w_{i-1}). \end{aligned} \tag{2.45}$$

From Equation 2.45 we can see that each term represents the probability of a word, conditioned on all previous words. In practice, however, it is not feasible to try to estimate these probabilities for long word sequences. The common solution to this problem is to approximate each term in Equation 2.45 by only considering the previous $N-1$ words, that is

$$P(w_n|w_1,\cdots,w_{n-1}) \approx P(w_n|w_{n-N+1},\cdots,w_{n-1}). \tag{2.46}$$

LMs that make this assumption are referred to as N-grams. The maximum likelihood estimate of these probabilities can be computed from a corpus of training text as follows

$$P(w_n|w_{n-N+1},\cdots,w_{n-1}) = \frac{C(w_{n-N+1},\cdots,w_n)}{C(w_{n-N+1},\cdots,w_{n-1})}, \tag{2.47}$$

where $C(\cdot)$ is the count.

The most obvious problem that occurs if we build a LM using Equation 2.47 is that all unseen N-grams will be assigned a zero probability. One method that has been devised for correcting this problem is discounting, which redistributes some of the probability mass of the observed N-grams to the unseen ones. Several different procedures have been proposed for determining how much probability mass to redistribute, including Absolute [13], Good-Turing [14], and Witten-Bell [15] discounting. Discounting is commonly combined with a procedure known as backing-off [16], which defines how to redistribute the probability mass.

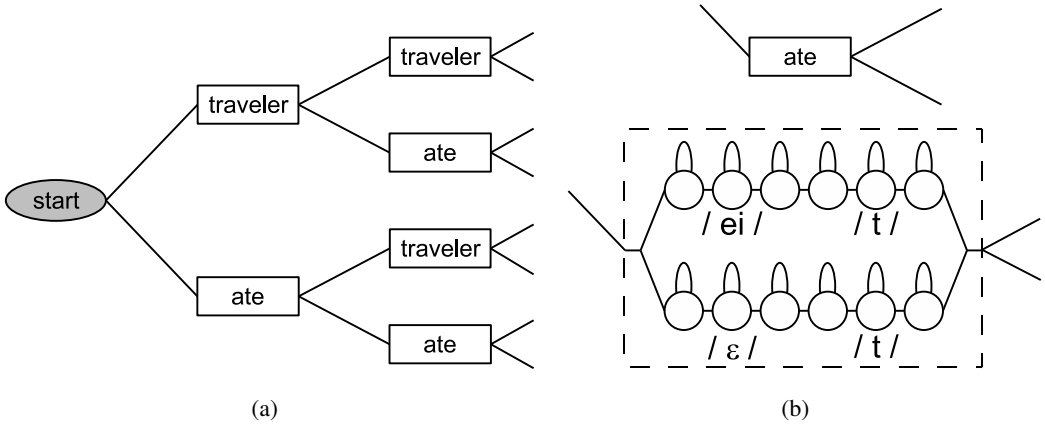


Figure 2.12: Word tree structure for the decoder (a) with a vocabulary limited to the words *traveler* and *ate*; (b) multiple pronunciations for the word *ate* are represented by including additional branches within the word node.

2.6 Decoder

The task of the decoder can be illustrated in a tree structure as shown in Figure 2.12. The tree is constructed by creating a start node with branches to every word in the vocabulary. This represents the first word in the sequence. Next, this process is repreated for each word; that is, another set of branches leading to all words in the vocabulary is created from each word. If we continue in this manner, all possible word sequences can be represented by tracing different paths through the tree. The decoder can be implemented using this tree structure by replacing each word with the corresponding HMMs. Multiple pronunciations for a single word can be represented by including additional branches within each word node. In addition, the branches connecting each word are associated with a language model probability.

The most strightforward method for obtaining the hypothesized word sequence would be to consider each path in the tree structure; that is, every word and HMM state sequence.[3] The quantity $P(O|W_i)$ can be calculated from the composite HMM by Equation 2.23 and the quantity $P(W_i)$ can be approximated by multiplying the language model probabilities associated with each inter-word branch. The hypothesized word sequence is simply the path that maximizes Equation 2.3. Even for small vocabularies, however, this method is not feasible in practice. One alternative decoding strategy that has been developed is the Token Passing Algorithm.

[3]Note that the same word sequence can be generated by many different HMM state sequences.

2.6.1 Token Passing Algorithm

This section discuss the Token Passing Algorithm [17], which is also referred to as Viterbi decoding. Consider first a single path through the tree structure. Recall that the the joint probability of the observation sequence O and HMM state sequence Q is given by Equation 2.22. In logarithmic form, this probability can be seperated into a sum of transition probabilities and a sum of state output probabilities

$$\log P(O, X|\lambda) = \sum_{t=0}^{T} \log a_{x(t)x(t+1)} + \sum_{t=1}^{T} \log b_{x(t)}(o_t). \tag{2.48}$$

In addition, recall that the probability of the word sequence W is given by Equation 2.45, which can be expressed in logarithmic form as follows

$$\log P(W) = \log P(w_1) + \log P(w_2|w_1) + \log P(w_3|w_1, w_2) + \ldots. \tag{2.49}$$

Suppose that we represent the path taken through the tree by moving a token from the start node through the HMM state sequence Q. Furthermore, this token is associated with a score that is calculated by summing the log state transition probabilities, log state output probabilities, and log LM probabilities asscociated with the path. The basic concept of the token passing algorithm is to propogate multiple tokens through different paths of the tree structure and select the token with the highest score.

The token passing algorithm can be formulated as follows. Initially, a token with a score of zero is placed at the start node. Next, this token is copied to all connecting states and, using the first observation vector, the score of each token is updated. This process is repeated for all observation vectors in the sequence; that is, all tokens are copied to all connecting states and the scores for each token are updated. When the entire observation sequence has been processed, the word ends are scanned and the token with the highest score is selected. The path of this token yields the hypothesized word sequence.

In practice, two modifications to the token passing algorithm are usually implemented. First, the log LM probabilities are typically modified as follows

$$\log P(w_i|w_i,\ldots,w_{i-1}) \Longrightarrow s \log P(w_i|w_i,\ldots,w_{i-1}) + p, \tag{2.50}$$

where s is the LM scale factor and p is the word insertion penalty. Typically, s is chosen to be greater than zero to give more emphasis to the LM and p is chosen to be less than zero to reduce the number of incorrectly inserted words in the hypothesized word sequence. Second, a procedure known as beam-pruning is typically applied after each observation vector is processed to decrease the search space. Beam-pruning deletes all tokens whose score is less than the best score minus some beam-width.

Chapter 3

Articulatory Feature Detection

Articulatory Features (AFs) describe the way in which the speech organs are used when producing speech sounds. The IPA scheme, which was introduced in Section 2.1, is one method that has been developed for classifying speech sounds usings AFs. Vowels are defined in terms of tongue height, tongue fronting, and whether the lip shape is rounded or unrounded. Consonants are defined by the place of articulation, manner of articulation, and whether the airflow from the lungs is voiced or unvoiced. To clarify, consider the phonemes /e/ and /b/ from Figures 2.3 and 2.4, respectively. The phoneme /e/ is described by the AFs *vowel*, *close-mid*, *front*, *unrounded*, and *voiced*;[1] the phoneme /b/ is described by the AFs *consonant*, *bilabial*, *plosive*, and *voiced*.

This chapter discusses AF detection in English, German, Spanish, and Japanese. Section 3.1 presents some of the past research done using AFs and discusses the contributions of this thesis. Section 3.2 discusses the corpora that were used and provides a brief overview of the phonetic and articulatory characteristics of the languages. Section 3.3 explains how the speech database was automatically transcribed at the articulatory level. Sections 3.4 and 3.5 describe the procedure used for training and evaluating the GMM- and MLP-based AF detectors developed in this thesis. Finally, Section 3.6 discusses the AF detection experiments performed and presents the results.

[1] Recall that all vowels are normally voiced.

3.1 Previous Work

The most straightforward method of obtaining articulatory information is to physically measure the movement of the speech organs. This can be accomplished by several different methods, including ultrasonography [18], using x-rays [19], and electromagnetic articulography [20]. In real world applications, however, these methods are impractical because they subject the user to strict physical constraints, such as exposure to an x-ray machine or wearing a helmet. Due to this limitation, articulatory information is usually extracted from the acoustic speech waveform. This can be accomplished by either designing a set of feature vectors that can be derived from the speech waveform or using classifiers to detect AFs from a set of acoustic feature vectors, such as MFCCs. The following discussion presents some of the research that has been done using AFs.

Abdelatty Ali et al. [21] derived a set of features for classifying stop consonants. Three features for voicing status and six features for detecting the place of articulation were extracted from the acoustic speech signal. A decision-tree-like algorithm was used to classify the speech sounds based on these features. The database used consisted of 1200 stop consonants from the TIMIT corpus (English speech). An accuracy of 96% was obtained for detecting voicing, and 90% for classifying the place of articulation.

Kocharov et al. [22] derived a set of articulatory motivated features including a voicedness and spectrum derivative feature. These features were combined with MFCCs by concatentaing the two streams and then performing Linear Discriminant Analysis (LDA). The features were evaluated on the SieTill corpus (German speech) and the Verbmobil II corpus (German speech). Incorporating the voicedness and spectrum derivative feature resulted in a 0.9% reduction in Word Error Rate (WER) on the SieTill corpus and a 0.8% reduction in WER on the Verbmobil II corpus.

Kirchhoff [23] used MLPs to detect five multi-valued articulatory features. Two different feature sets were used as input to the classifiers: in clean speech log - Relative Spectral Transform - Perceptual Linear Prediction (RASTA-PLP) features were used and Modulation Spectrogram (MS) features were used in noisy speech. The outputs of the detectors were combined using a second higher-level MLP that generated a vector of phoneme probabilities, which was passed on to an HMM-based decoder. This system was compared to a RASTA-PLP- and MS-based speech recog-

nizer on the OGI Numbers95 corpus (American English Speech). The proposed system increased the WER by 0.5% in clean speech and decreased the WER by up to 6.8% in noisy speech. Articulatory feature detectors were also created on the Verbmobil corpus (German speech) and the outputs of the detectors were used as the input to a vector quantization based HMM system. When merged with a MFCC-based speech recognizer, a reduction in WER of up to 1.6% was obtained on clean speech.

Scharenborg et al. [24] compared the performance of MLP and SVM classifiers on the task of detecting seven multi-valued articulatory features. MFCC features were used as the input to the classifiers. On the TIMIT corpus, the classification accuracies obtained with the MLPs ranged from 19.7% to 93.8%, and with the SVMs the classification accuracies ranged from 12.5% to 91.9%. The authors showed that while the MLP classifiers outperformed the SVM classifiers, both detectors tend to make similiar errors.

Stüker [25] used MFCC-based GMMs to detect 33 different articulatory features. The Wall Street Journal corpus and the GlobalPhone corpus were used to develop detectors in English, German, Spanish, Japanese, and Chinese-Mandarin. Both monolingual (trained with only one language) and multilingual detectors (trained with all languages) were created. The average classification accuracy obtained with the monolingual detectors ranged from 92.9% to 95.2% when evaluated on the same language that was used for training, and from 83.2% to 89.2% when evaluated across different langauges. Using the multilingual detectors, the average classification accuracy ranged from 88.7% to 90.9%. A speech recognizer was developed that combined the scores from the GMM detectors with a MFCC-based context-dependent HMM at the log-likelihood level. Reductions in WER of up to 2.0% were obtained using the monolingual detectors and up to 1.4% using the multilingual detectors.

Morris et al. [26] used MLPs with Perceptual Linear Prediction (PLP) input features to detect five multi-valued phonetic attribute features (sonority, voice, manner, place, tongue height, tongue fronting, and tense). The outputs of the classifiers were processed with a Karhunen-Loéve Transform (KLT) and used as the feature set for an HMM- and Conditional Random Field (CRF)-based phone recognizer. Compared to a PLP-based triphone HMM system on the TIMIT corpus, the

phonological features yielded an increase in accuracy of 1.40% with monophone HMMs, 6.61% with triphone HMMs, and 5.15% with monophone CRFs.

Frankel et al. [27] compared the performance of Dynamic Bayesian Networks (DBN) and Artificical Neural Networks (ANN) on the task of classifying six multi-valued articulatory features. The TIMIT corpus was used for all evaluations. An average classification accuracy of 81.5% was obtained using the DBNs, compared to an average classification accuracy of 85.7% obtained using the ANNs.

In this thesis, GMM-based AF detectors were created for English, German, Spanish, and Japanese. The same procedure as described in [25] was used to train and evaluate the detectors, with the following exceptions. First, the phoneme alignments generated in this thesis were obtained using three state context-independent HMMs, whereas Stüker used three state context-dependent HMMs. Second, in [25] the GMM-based AF detectors were trained and evaluated using only those frames of speech time-aligned with the middle states of each HMM—in this thesis all frames of speech were used. Third, five additional AFs (*near-front*, *near-back*, *near-open*, *mid*, and *near-close*) were defined to uniquely represent all vowels on the IPA chart. Lastly, the GMM-based AF detectors in this thesis were created using a 39-dimensional MFCC feature set, whereas Stüker used 32-dimensional MFCCs.

MLPs have been shown by Kirchhoff [23] to be useful for detecting AFs in English and German. In this thesis, MLP-based AF detectors were developed for English, German, Spanish, and Japanese; however, several modifications were made to the procedure descibed in [23] to more directly compare the performance of the GMM- and MLP-based AF detectors. First, a more extensive AF set was defined in this thesis: 15 additional AFs were used to represent English speech sounds and 10 additional AFs were used to represent German speech sounds. Second, Kirchhoff used five multi-valued AF detectors, whereas in this thesis binary detectors were created for each AF. Third, MFCC features were used as the input to all AF detectors created in this thesis. Lastly, multilingual AF detectors were also created using speech data from all four languages.

The outputs of the AF detectors were used as the feature set for an HMM-based speech recognizer. The same procedure as descibed in [26] was used to create the AF-based feature sets, except that delta coefficients were also included. Note that although the feature set was created in a sim-

ilar manner, the scores were generated using a different procedure and additional languages were considered.

3.2 Database Overview

This section provides a brief overview of the database used for the experiments discussed in this thesis. Sections 3.2.1 and 3.2.2 provide a summary of the Wall Street Journal and GlobalPhone corpora, respectively. Section 3.2.3 describes how training, development, and test partitions of the database were created. Finally, Section 3.2.4 summarizes the phonetic and articulatory characteristics of the languages considered.

3.2.1 Wall Street Journal Corpus

Phase II of the Wall Street Journal Continuous Speech Recognition (WSJ1-CSR) corpus [28] was used for English speech. Collection of the WSJ1-CSR corpus began in 1992 and was completed in 1993 by SRI International. This corpus includes both read and spontaneous English speech spoken by non-journalist and journalist subjects. All read text was selected from Wall Street Journal articles and the spontaneous speech was produced by journalist subjects who improvised news articles on pre-selected topics. Most of the recordings were performed in quiet rooms and two microphones were used: a Sennheiser close-talking microphone and several different secondary microphones. Speech data were recorded at a 16 kHz sampling rate and 16 bit resolution. WSJ1-CSR includes approximately 78,000 training utterances ($\sim$73 hours of speech) and 8,200 test utterances ($\sim$8 hours of speech) spoken by 75 different speakers. Approximately 4,000 of the training utterances and 6,800 of the test utterances are spontaneously produced utterances. All speech data are transcribed at the utterance level; that is, only the time-alignments between the acoustic waveform and the sentence spoken is known.[2]

[2]Few speech corpora are available that provide word and phoneme time-alignments.

3.2.2 GlobalPhone Corpus

GlobalPhone [29] is a multilingual text and speech database that covers 15 different languages: Arabic, Chinese-Mandarin, Chinese-Shanghai, Croatian, Czech, French, German, Japanese, Korean, Brazilian-Portuguese, Russian, Spanish, Swedish, Tamil, and Turkish. Collection of the GlobalPhone corpus began in 1995 and was completed in 1998 at the Karlsruhe University. This corpus consists of read text selected from national newspapers covering national and international political news and economic news. The recordings were done in quiet rooms using a Sennheiser close-talking microphone and the subjects were native speakers living in their home country. Speech data were originally recorded at a 48 kHz sampling rate and 16 bit resolution and downsampled to a 16 kHz sampling rate. In total, GlobalPhone includes over 112,000 utterances ($\sim$ 270 hours of speech) spoken by over 1300 different speakers. The speech data are transcribed at the utterance level.

3.2.3 Training, Development, and Test Sets

The WSJ1-CSR and GlobalPhone corpora were used to form training, development, and test [3] sets covering four different languages: English, German, Spanish, and Japanese. Only read text produced by non-journalist subjects using the Sennheiser microphone was used from the WSJ1-CSR corpus. This was done in an effort to match the GlobalPhone database as closely as possible to minimize the effects of recording equipment and speech type (*i.e.*, read, spontaneous, or conversational) on recognition performance. To ensure that each language had a comparable amount of speech data, only 100,000 utterances were selected from the WSJ1-CSR corpus.

The speech data from each language were partitioned as follows: 80% of the speakers were used for training, 10% for the development, and the remaining 10% for the test set. An attempt was made to evenly distribute the male and female speakers within each partition. Table 3.1 shows the database statistics for the training, development, and test sets. It should be noted that there is a relatively small amount of speech data available for each language. In comparison, commercial speech recognizers are typically trained with hundreds of hours of speech data.

[3] Note that the test set is not used in any experiments discussed in this thesis as it is reserved for future evaluations.

Table 3.1: Overview of the training, development, and test sets formed from the GlobalPhone and WSJ1-CSR corpora for English (EN), German (GE), Spanish (SP), and Japanese (JA). Spkrs is the number of different speakers and M/F gives the distribution of male and female speakers.

	Training			Development			Test		
	Hours	Spkrs	M/F	Hours	Spkrs	M/F	Hours	Spkrs	M/F
EN	17.82	80	40/40	2.25	10	5/5	2.15	10	5/5
GE	14.04	65	60/5	2.11	6	5/1	2.18	6	5/1
SP	17.32	84	38/46	2.28	8	3/5	1.74	8	3/5
JA	25.01	116	85/31	3.69	12	9/3	3.14	11	8/3

3.2.4 Phoneme and AF Inventory

This section provides an overview of the phoneme and AF inventory for each language. Table 3.2 shows the phoneme set for each language; the last column in the table sums the number of unique phonemes across the languages. The number of phonemes shared across languages is as follows: 16 are shared by all four languages, 23 are shared by at least three languages, 37 are shared by at least two languages, and the remaining 39 only appear in a single language. The number of phonemes required to represent each language varies from 30 for Japanese to 44 for English.

The majority of the phonemes listed in Table 3.2 can be found in Figures 2.3 and 2.4, however, several additional symbols are required to form the complete phoneme set. A supplement to the IPA consonant chart is given in Table 3.3 that includes the additional place of articulation *labialvelar* and manner of articulation *affricate*. The place feature labialvelar represents a speech sound that has two places of articulation, namely the velum and the lips; an affricate is a speech sound that begins as a stop[4] and ends as a fricative. In addition to the consonants listed in Table 3.3, several IPA markings, or diacritics, are also needed. These diacritics are used to represent length, stress, and rhoticity. Length is simply the relative duration of a speech sound and is typically categorized as either long or short. Long sounds are represented in the IPA scheme by placing a length marking ː after the symbol. Stress refers to the relative emphasis of a speech sound and is characterized by the combined effects of pitch, duration, and loudness. Primary stress is represented in the IPA scheme placing a high vertical line ˈ before the symbol. Vowel retroflexion, or rhoticity, is achieved

[4] A stop is a superset of plosives; plosives are stops produced with the restriction that the velum must be raised to prevent airflow from escaping through the nasal cavity.

Table 3.2: IPA phoneme set for English, German, Spanish, and Japanese.

	Language				
Phonemes	English	German	Spanish	Japanese	$\sum$
p,b,t,d,k,g,m,n,s,z,j,l,f	x	x	x	x	13
i,e,o	x	x	x	x	16
ʧ	x		x	x	17
ɑ	x		x	x	18
ŋ	x	x	x		19
u	x	x	x		20
ʃ,ʦ,h	x	x		x	23
ɾ,θ,ð	x		x		26
r,x		x	x		28
v	x	x			29
ə,ɛ,aɪ,aʊ	x	x			33
eː,iː,oː		x		x	36
ʤ	x			x	37
ʔ,ɴ,ʊ				x	40
ɯ,ɯː				x	42
ɲ,β,ɤ,ʎ			x		46
ˈɑ,ˈe,ˈi,ˈo,ˈu,ɑi,ɑu,ei,eu,oi			x		56
ʒ,ɹ	x				58
ɨ,ɪ,ʊ,ʌ,ɔ,æ,ɔɪ,ɚ,ɝ,w	x				68
ç		x			69
y,ø,ɐ,a,ɔʏ,øː,yː,uː		x			76
Total	44	40	39	30	

Table 3.3: Additional IPA consonants.

	Dental	Postalveolar	Labialvelar
Affricate	ʦ	ʧ ʤ	
Approximant			w

Table 3.4: AFs for consonants and vowels; each AF is assigned a number.

CONSONANTS (0)		
Place	Manner	Voicing
bilabial (1)	plosive (12)	voiced (20)
labiodental (2)	nasal (13)	unvoiced (21)
labialvelar (3)	trill (14)	
dental (4)	tap or flap (15)	
alveolar (5)	fricative (16)	
postalveolar (6)	approximant (17)	
retroflex (7)	lateral approximant (18)	
palatal (8)	affricate (19)	
velar (9)		
uvular (10)		
glottal (11)		

VOWELS (22)		
Tongue Height	Tongue Fronting	Lip Shape
close (23)	front (30)	rounded (35)
near-close (24)	near-front (31)	unrounded (36)
close-mid (25)	central (32)	
mid (26)	near-back (33)	
open-mid (27)	back (34)	
near-open (28)		
open (29)		

by curling the tongue tip while maintaining the basic tongue position for a particular vowel and is represented in the IPA scheme by appending the marking ˞ to the symbol.

The AFs required to represent the consonants and vowels for all four languages are shown in Table 3.4. The numbers in parentheses after each AF is the feature number. In addition, an AF for stress (37) and silence (38) were also defined. The AF set for each language was defined using the same procedure as in [25], except that several additional AFs are defined for tongue position that are not labeled on the IPA chart—namely, *near-front*, *near-back*, *near-open*, *mid*, and *near-close*. These are used to represent the symbols that do not lie at intersecting points of the vertical and horizontal axis. The total number of unique AFs across the languages is 39. The number of AFs shared across languages is as follows: 23 are shared by all four languages, 24 are shared by at least three languages, 31 are shared by at least two languages, and the remaining seven only appear in a single language. The number of AFs required to represent each language varies from 26 for Japanese to 36 for English.

The percentage of AFs shared across languages is greater than the percentage of phonemes shared: 82% of the AFs are shared by at least two languages, whereas only 49% of the phonemes appear in multiple languages. From this perspective, modeling speech at the articulatory level instead of at the phoneme level could be advantageous when building speech recognizers for multiple languages. This increase in unit sharing could reduce the amount of transcribed speech data that is needed when training a recognizer.

3.3 AF Alignments

In the previous section, it was mentioned that the WSJ1-CSR and GlobalPhone corpora used in this thesis are only transcribed at the utterance level. The first resource that is needed to create AF detectors, however, is a speech database that is transcribed at the AF level. Ideally the desired time-alignments would be manually produced by a linguist; however, this is a very time consuming and expensive process. As an alternative, the alignments were produced by automatic means. The following discussion describes the procedure used.

Context-independent HMM phoneme models were created for each of the languages using the Hidden Markov Model ToolKit (HTK) [9]. The procedure discussed in Section 2.4 was used to train the models. Phonemes were modeled using three emitting states, each of which included 32 mixture components with diagonal covariance matrices. The feature set consisted of 12 MFCCs calculated every 10ms using 20ms frames. Energy, delta, and acceleration coefficients were also included to form a 39-dimensional vector. The HMMs were used to phonetically align the training and development data using the Viterbi Algorithm, as discussed in Section 2.3.3. Note that this procedure is similar to that described in [25], except that (1) a different MFCC feature set was used, and (2) context-independent HMMs were used instead of context-dependent HMMs. Lastly, the phoneme-level alignments were mapped to the corresponding AF representation. For example, all frames of speech aligned with the phoneme /b/ were labeled as *consonant*, *bilabial*, *plosive*, and *voiced*. All frames of speech that were time-aligned with diphthongs, which do not have a single target configuration, were ignored.

3.4 GMM-based AF Detectors

The task of the AF detectors developed in this thesis is to detect the presence of AFs in a spoken utterance. This section describes the procedure used for training and evaluating Gaussian Mixture Model (GMM)-based AF detectors. Recall the definition of a GMM, which is of the form

$$p(o) = \sum_{m=1}^{M} c_m \mathcal{N}(o; \mu_m, \Sigma_m), \tag{3.1}$$

where c_m is the weight of the m^{th} mixture component, M is the total number of mixtures, and $\mathcal{N}(o; \mu, \Sigma)$ is a single Gaussian density given by Equation 2.15. Denote the complete model as follows

$$\theta = \{(c_m, \mu_m, \Sigma_m) \,|m = 1, 2, \cdots, M\} \tag{3.2}$$

Consider first the problem of given an observation sequence $O = o_1, o_2, \ldots, o_T$ and a model θ, how do we update the model parameters such that the probability of the given observation sequence is maximized? The most common solution to this problem is the Expectation Maximization (EM) Algorithm [11]. The EM Algorithm is an iterative procedure that estimates new model parameters $\hat{\theta}$ at each iteration such that $p(O|\hat{\theta}) \geq p(O|\theta)$; θ is used here to denote the model parameters estimated from the previous iteration. The equations for updating the mixture weights, means, and covariances as follows [30]

$$\begin{aligned}
\hat{c}_m &= \frac{1}{T} \sum_{t=1}^{T} p(m|o_t, \theta) \\
\hat{u}_m &= \frac{\sum_{t=1}^{T} p(m|o_t, \theta) o_t}{\sum_{t=1}^{T} p(m|o_t, \theta)} \\
\hat{\Sigma}_m &= \frac{\sum_{t=1}^{T} p(m|o_t, \theta) \left(o_t - \hat{u}_m\right) \left(o_t - \hat{u}_m\right)^T}{\sum_{t=1}^{T} p(m|o_t, \theta)},
\end{aligned} \tag{3.3}$$

where the probability of mixture component m for the given observation o_t and model θ is calculated as

$$p(m|o_t, \theta) = \frac{c_m \mathcal{N}(o_t; \mu_m, \Sigma_m)}{\sum_{i=1}^{M} c_i \mathcal{N}(o_t; \mu_i, \Sigma_i)}. \tag{3.4}$$

Note that in order to perform the first iteration of the EM algorithm an initial model θ is needed. The GMMs used in this thesis were initialized by the following procedure. First, a mean vector was randomly generated for each of the M mixtures. Next, one iteration of k-means clustering was performed and the initial mixture weights, means, and covariances for each mixture were calculated based on the resulting clusters.

The procedure used for creating the GMM-based AF detectors was based on the work of Stüker, which can be summarized as follows. Consider first a single AF and denote the presence of the AF as f and the absence of the AF as $\overline{f}$. One method for classifying the speech vector o is to declare the AF present if $P(f|o) > P(\overline{f}|o)$ and declare the AF absent otherwise. This inequality can be expressed in logarithmic form as follows

$$\log \frac{P(f|o)}{P(\overline{f}|o)} > 0. \tag{3.5}$$

By applying Baye's Rule, the left-hand side of Equation 3.5 can be expressed as

$$\log \frac{P(f|o)}{P(\overline{f}|o)} = \log P(o|f) + \log P(f) - \log P(o|\overline{f}) - \log P(\overline{f}). \tag{3.6}$$

The probabilities $P(f)$ and $P(\overline{f})$ can be estimated from the training corpus by calculating the relative frequency of the AF. The terms $P(o|f)$ and $P(o|\overline{f})$ can be estimated using GMMs. In [25] the AF detectors were trained using only speech data that was time-aligned with the middle states of the HMM phoneme models. This was to account for the lack of manually transcribed data and to reduce the effects of coarticulation. Preliminary experiments, however, suggested that using all speech data could lead to an improvement in system performance in the paradigm of this thesis. Therefore, the feature present f GMM was trained using speech data that was time-aligned with the AF and the feature absent $\overline{f}$ GMM was trained with all remaining speech.

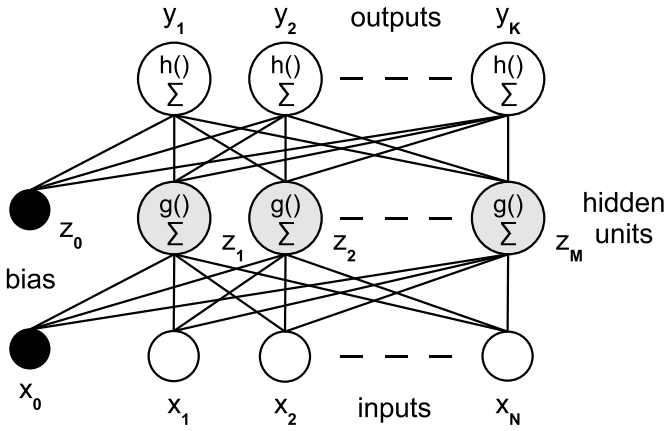


Figure 3.1: MLP neural network with a single layer of hidden units; there are a total of N input nodes, M hidden nodes, and K output nodes; the variables $\{x_i\}$, $\{z_i\}$, and $\{y_i\}$ represent the outputs of the nodes. The hidden nodes compute a weighted sum of their inputs, which is passed through a non-linear activation function $g(\cdot)$; similarly, the output nodes compute a weighted sum of their inputs, which is passed through a non-linear activation function $h(\cdot)$.

The GMM software package from the Massachusetts Institute of Technology Lincoln Laboratory (MIT-LL) [30] was used to train and evaluate the GMMs created in this thesis. All models included 256 mixture components and used diagonal covariance matrices. Training was performed using 15 iterations of the EM Algorithm. The feature set was the same as descibed in Section 3.3, and all features were generated using HTK.

3.5 MLP-based AF Detectors

In addition to the GMM-based detectors discussed in the previous section, Multi-Layer Percepton (MLP)-based detectors were also created for each AF. An MLP is a neural network consisting of a single layer of input nodes, one or more layers of hidden nodes, and a single layer of output nodes. Figure 3.1 shows the general structure of an MLP with one layer of hidden units; there are a total of N input variables $\{x_i\}$, M hidden nodes with outputs $\{z_i\}$, and K output variables $\{y_i\}$. In addition, there are two bias nodes defined as $x_0 = 1$ and $z_0 = 1$. Connections run from each unit in one layer to every unit in the next layer.

The network shown in Figure 3.1 can be expressed as follows [31]. First, each hidden node computes a weighted sum of the input variables and a bias. Denote the sum for the j^{th} hidden node

as a_j, then

$$a_j = \sum_{i=0}^{N} w_{ij}^{(1)} x_i, \tag{3.7}$$

where $w_{ij}^{(1)}$ denotes the weight associated with the connection running from the i^{th} input node to the j^{th} hidden unit; when $i = 0$ the weight represents the bias. Next, the output of the j^{th} hidden node is obtained by applying an activation function $g(\cdot)$ to the sum calculated by Equation 3.7, that is $z_j = g(a_j)$.

The output variables are calculated in a similar manner. Each output node computes a weighted sum of the quantities calculated by the hidden nodes and a bias. Denote the sum for the k^{th} output node as b_k, then

$$b_k = \sum_{j=0}^{M} w_{jk}^{(2)} z_j, \tag{3.8}$$

where $w_{jk}^{(2)}$ denotes the weight associated with the connection running from the j^{th} hidden node to the k^{th} output node; when $j = 0$ the weight represents the bias. The output variable y_k is obtained by applying an activation function $h(\cdot)$ to the sum calculated by Equation 3.8, that is $y_k = h(b_k)$. The entire network can be expressed as follows

$$y_k = h\left(\sum_{j=0}^{M} w_{jk}^{(2)} g\left(\sum_{i=0}^{N} w_{ij}^{(1)} x_i\right)\right). \tag{3.9}$$

The task of an MLP classifier is to assign a feature vector $X = x_1, x_2, \cdots, x_N$ to one of K classes; denote the individual classes as C_k. This can be accomplished by a network with K output nodes where each output y_k is associated with the class C_k. Ideally, if we are given a feature vector that belongs to class C_k the quantity y_k should be one and all other output variables should be zero.

Consider the problem of given a feature vector X that belongs to class C_k, how do we update the weights of the network? One method for accomplishing this is the gradient descent technique, which can be summarized as follows. Note that since the desired output of the network is known the error of the system E can be calculated. Assuming that the error is a differentiable function, the weights can be updated as follows

$$\Delta w_{nm} = -\eta \frac{\partial E}{\partial w_{nm}}, \tag{3.10}$$

where η is a learning rate parameter. Equation 3.10 can be interpreted as follows. Consider a network where the error of the system can be written as a function of the weights; denote the complete set of weights as Ω and the error function as $E(\Omega)$. Assuming that $E(\Omega)$ is a differentiable function, this function decreases fastest in the direction opposite the gradient vector, that is, in the direction $-\nabla E(\Omega)$. Thus, if we adjust the weights of the network by moving a small distance in Ω-space in the direction $-\nabla E(\Omega)$, the error of the system will reduced.

The International Computer Science Institute (ICSI) QuickNet software package [32] was used to train and evaluate the MLPs created in this thesis. A single MLP was trained for each AF to classify the feature as either present or absent. The same feature set as descibed in Section 3.3 was used as the input, except that the MFCC features were first normalized by subtracting the mean value of coefficient and dividing by the standard deviation. All features were generated using HTK. All networks included 100 hidden units that used the logistic function as the activation function, which is given as

$$g(a_j) = \frac{1}{1 + \exp(-a_j)}. \tag{3.11}$$

The output activation function used was the softmax function, which is defined as follows

$$h(b_k) = \frac{\exp(b_k)}{\sum_{n=1}^{K} \exp(b_n)}. \tag{3.12}$$

The main differences between this procedure and that described in [23] are (1) MFCC features were used as the input for all detectors, and (2) binary MLPs were created for each AF instead of multi-valued AF detectors. This was done in order to more directly compare the performance of the GMM- and MLP-based AF detectors.

In addition to the detectors described above, a second set of AF detectors that used a context window were created. MLPs created without a context window use the speech vector o_t as input to classify the speech frame at time t; MLPs with a context window of length $2L + 1$ use the speech vectors $o_{t-L}, \cdots, o_{t+L}$ as input to classify the speech frame at time t. All networks were trained with a context window of length nine and used the same hidden and output layer as described above; that is, only the input layer was modified. In the remainder of this thesis, the detectors trained

without a context window will be referred to as MLP-based and the detectors trained with a context window will be referred to as MLPwC-based.

3.6 AF Detection Experiments

This section discusses the AF detection experiments performed and presents the results obtained. Section 3.6.1 introduces the performance metrics that were used to evaluate the detectors. Section 3.6.2 describes the monolingual experiments. The term *monolingual* is used here to refer to experiments that were performed on a single language; that is, the AF detectors were trained and evaluated on the same language. Finally, Section 3.6.3 describes the multilingual AF detection experiments; *multilingual* is used here to refer to detectors that were trained on multiple languages.

3.6.1 Performance Metrics

The metrics used to evaluate the performance of the AF detectors are accuracy and Equal Error Rate (EER). Recall from Sections 3.4 and 3.5 that the detectors developed in this thesis classify speech vectors extracted every 10ms; that is, the speech signal is classified at the frame level. Therefore, the performance of the AF detectors is also evaluated at the frame level. Accuracy is defined as the percentage of correctly classified frames

$$\text{accuracy} = \frac{\text{number of correctly classified frames}}{\text{total number of frames}} \times 100\%. \tag{3.13}$$

A disadvantage of measuring detector performance using accuracy is that the distribution of the AFs is not taken into account. To clarify, consider the English AF *labiodental* which is only present in $\sim$3% of the frames. A labiodental detector that always classifies the AF as absent will have an accuracy of $\sim$97%; thus accuracy may not always be representive of the detection performance.

Two types of errors that a detector can produce are misses and false alarms. In the context of this thesis, a *miss* is when a detector fails to recognize an AF, and a *false alarm* is when a detector incorrectly classifies an AF as present. Miss probability is calculated as the percentage of frames where an AF is present that are incorrectly classified, and false alarm probability is the percentage

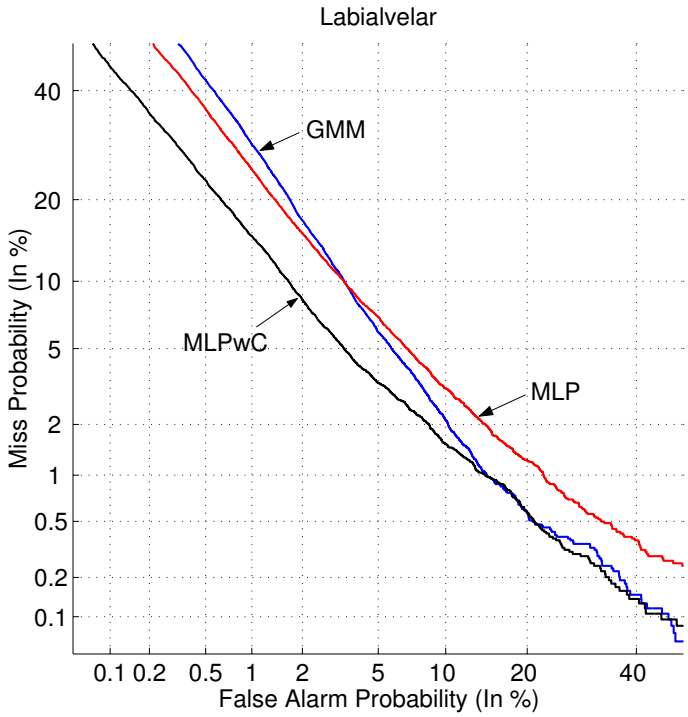


Figure 3.2: DET curves for the GMM-, MLP, and MLPwC-based labialvelar detectors evaluated on the English development partition.

of frames where an AF is absent that are incorrectly classified. EER is the error rate when the miss and false alarm probability are equal. A Detection Error Tradeoff (DET) curve is one method for plotting these errors for a detector with multiple operating points [33]. As an example, Figure 3.2 shows a DET plot for the English AF labialvelar. The EERs and DET curves presented in this thesis were computed using software available from the National Institute of Standards and Technology [34].

Consider first the DET curve for the GMM-based labialvelar detector. Recall that the GMM-based detectors classify AFs as either present or absent according to Equation 3.5; if the left-hand side of Equation 3.5 is greater than zero the AF is classified as present, and absent otherwise. The miss and false alarm probability of this detector can be calculated and plotted as a single point in Figure 3.2. A DET curve is created by evaluating the miss and false alarm probability of the detector by varying the right-hand side of Equation 3.5 to values other than zero.

The DET curve for the MLP-based labialvelar detector was created in a similar manner by replacing the left-hand side of Equation 3.5 with the score output from the MLP-based detector.

This score was calculated by subtracting the output of the absent unit from the output of the present unit. It should be noted that the output activation function was removed prior to computing the scores so that the score distribution more closely approximates a Gaussian. The same procedure was used to create the DET curve for the MLPwC-based detector.

3.6.2 Monolingual AF Detection

One set of GMM-, MLP-, and MLPwC-based detectors were created for each language using the procedures discussed in Sections 3.4 and 3.5; thus if an AF exists in all four languages a total of 12 detectors were created for that AF. The development partition of each language was used to evaluate the performance of the detectors, where the reference transcriptions were generated using the procedure discussed in Section 3.3. It is important to note that since the reference transcriptions were generated by automatic means, the results presented in this section may only give an estimate of the true performance of the detectors.

Accuracy

The accuracy of the GMM-, MLP-, and MLPwC-based AF detectors for each language is shown in Figure 3.3. The error bars indicate the minimum and maximum accuracy of the individual AF detectors; the bottom of each box is the first quartile, the middle line is the median, and the top is the third quartile. The complete results can be found in Appendix A.

Overall, the GMM-based detectors yield the worst performance. The median accuracy for each language is as follows: 95.39% on English, 92.98% on German, 93.56% on Spanish, and 94.54% on Japanese. From Figure 3.3 we can see that roughly 75% of the AFs in each language are correctly classified at least 90% of the time and about 25% of the AFs are correctly classified at least 95% of the time. The lowest accuracy across all languages is 80.35% for the German AF *alveolar* and the maximum is 99.49% for the Spanish AF *affricate*. In comparison, Stüker [25] reported a mean classification accuracy of 93.83% on English, 92.94% on German, 93.46% on Spanish, and 95.22% on Japanese; the mean classification accuracies of the GMM-based AF detectors created in this thesis are 94.56% on English, 92.71% on German, 93.55% on Spanish, and

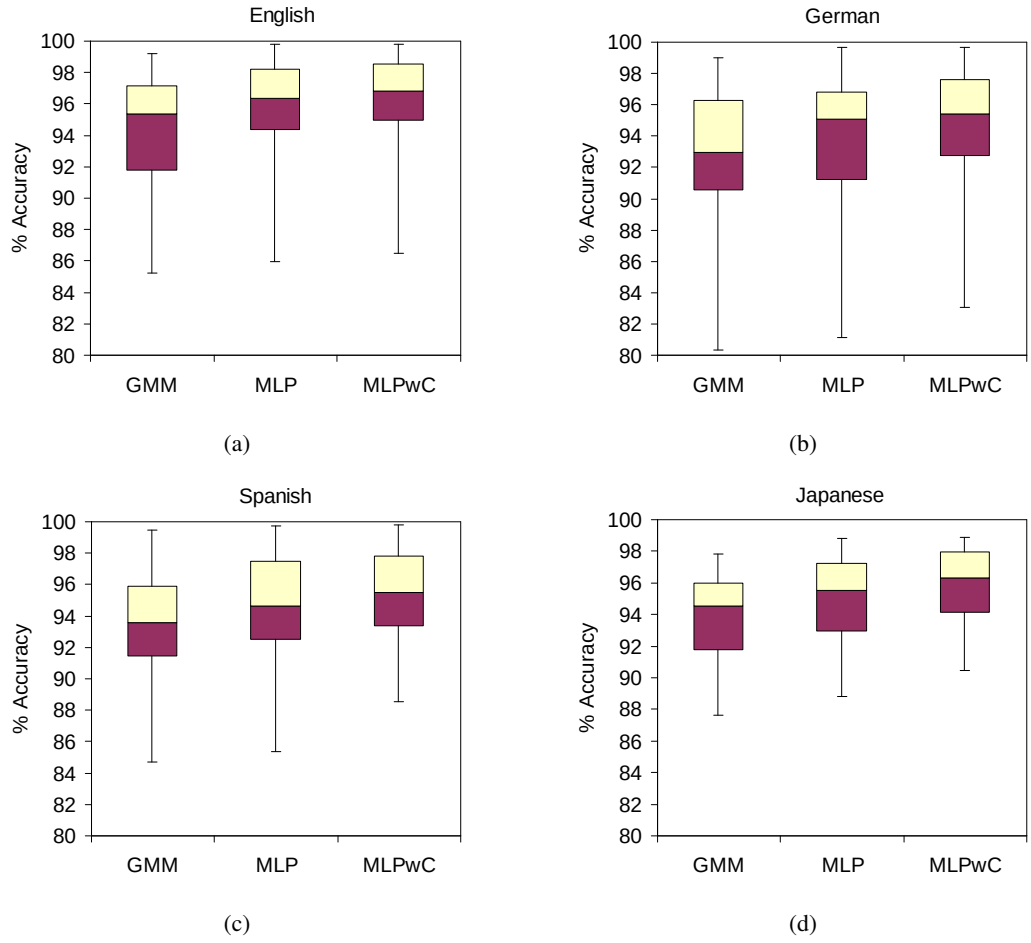


Figure 3.3: Accuracy rates of the monolingual GMM-, MLP-, and MLPwC-based AF detectors on the development partition.

93.77% on Japanese. It should be emphasized, however, that in [25] the GMM-based AF detectors were trained and evaluated using only speech frames that were time-aligned with the middle states of phoneme models, whereas in this thesis all frames of speech were used.

The MLP-based detectors yield higher accuracy rates than the GMM-based detectors for most AFs. The median accuracy increased by 0.97% on English, 2.08% on German, 1.05% on Spanish, and 0.98% on Japanese with the MLP-based detectors[5]. The maximum increase in accuracy is 4.45% for the German AF *voiced* and the maximum decrease is 1.10% for the German AF *unrounded*. It should be noted that the MLP-based detectors decreased the accuracy for only eight of

[5]All changes in performance presented in this thesis are given in absolute percentages, rather than relative improvements.

the 120 monolingual AFs; from Figure 3.3 it can be seen that the overall statistics improved for each language.

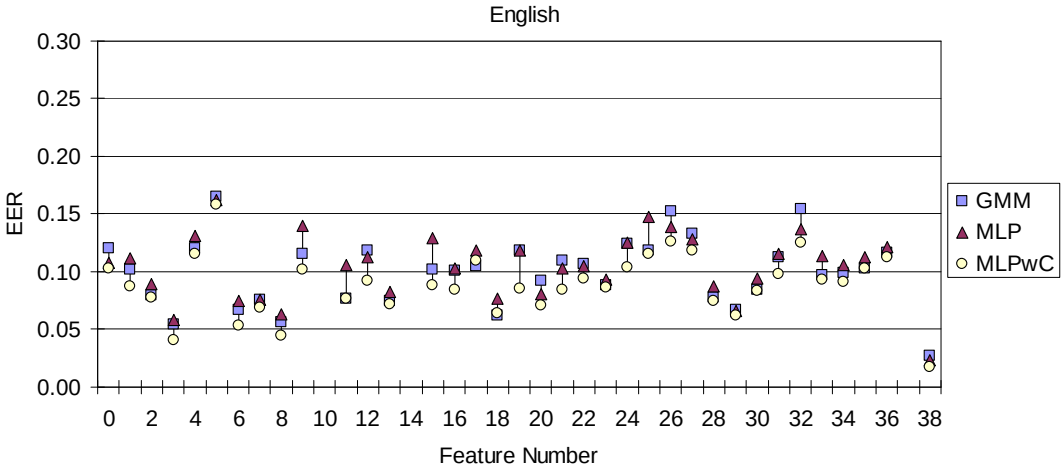
The MLPwC-based detectors outperformed the GMM- and MLP-based detectors for 118 of the 120 monolingual AFs. Compared to the MLP-based detectors, the median accuracy increased by 0.47% on English, 0.34% on German, 0.88% on Spanish, and 0.80% on Japanese. The maximum increase in accuracy is 3.17% for the Japanese AF *consonant* and the maximum decrease is 0.37% for the English AF *rounded*.

EER

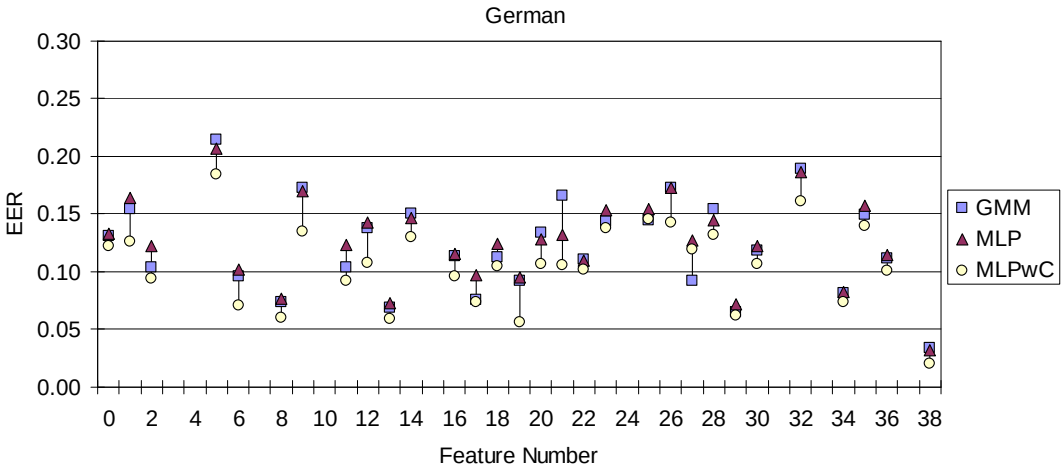
The EERs of the individual AF detectors for each language are shown in Figures 3.4 and 3.5. The feature numbers along the horizontal axis correspond to those given in Table 3.4. Consider first the GMM-based detectors; the median EER for each language is as follows: 10.19% on English, 11.60% on German, 11.27% on Spanish, and 9.71% on Japanese. The minimum EER is across all languages is 2.69% for the English AF *silence* (38) and the maximum is 21.43% for the German AF *alveolar* (5).

The MLP-based detectors yield lower EERs than the GMM-based detectors for 45 of the 120 monolingual AFs. An overall improvement in detection performance is only obtained for Japanese. The median increase in EER with the MLP-based detectors is 0.48% on English, 0.29% on German, and 1.39% on Spanish; the median decrease in EER on Japanese is 0.25%. The maximum reduction in EER is 3.38% for the German AF *unvoiced* (21) and the maximum increase is 4.12% for the Spanish AF *dental* (4).

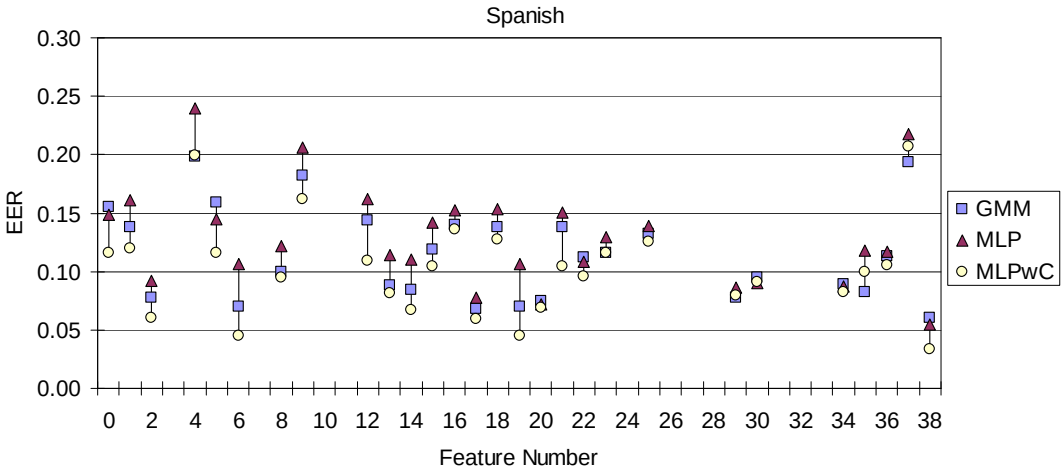
The MLPwC-based detectors outperform the GMM- and MLP-based detectors for 111 of the 120 monolingual AFs. Compared to the best GMM- or MLP-based detector for each AF, the median decrease in EER is 0.73% on English, 1.17% on German, 0.94% on Spanish, and 1.68% on Japanese. The maximum reduction in EER is 5.27% for the Japanese AF *glottal* (11) and the maximum increase is 2.70% for the German AF *open-mid* (27). Recall that the only difference between the MLP- and MLPwC-based detectors is the incorporation of context. From Figures 3.4 and 3.5 we can see that incorporating context may be more useful for some AFs than others. For example, in all



(a)



(b)



(c)

Figure 3.4: EERs of the individual GMM-, MLP-, and MLPwC-based monolingual AF detectors in (a) English (b) German and (c) Spanish on the development partition. The feature numbers correspond to those given in Table 3.4

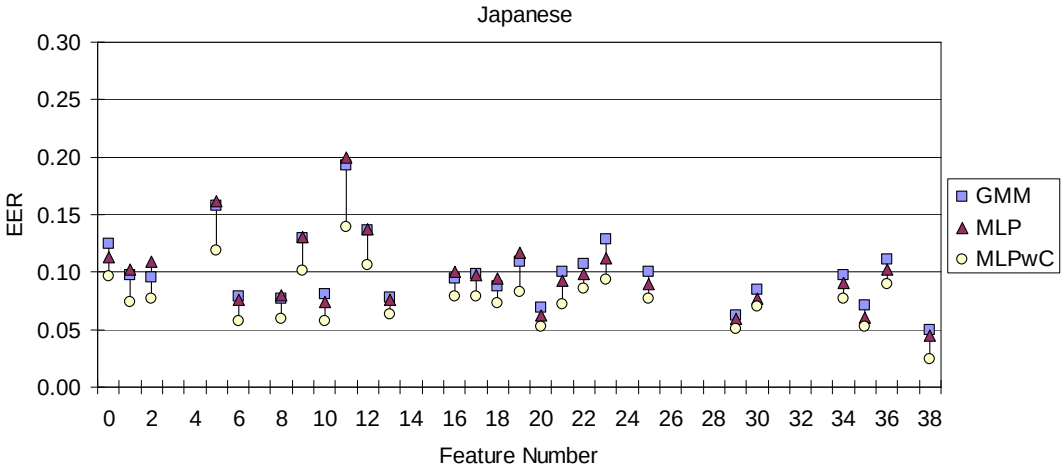


Figure 3.5: EERs of the individual GMM-, MLP-, and MLPwC-based monolingual AF detectors in Japanese on the development partition. The feature numbers correspond to those given in Table 3.4.

four languages a decrease in EER of at least 2.93% is obtained with the MLPwC-based detectors for the AFs *velar* (9) and *affricate* (19), whereas the maximum reduction in EER for the AFs *vowel* (22) and *open* (29) is 1.42%. Furthermore, the importance of context for detecting some AFs may be language dependent. For example, the MLPwC-based detectors for the AF *alveolar* (5) reduce the EER by 0.44% on English, 2.18% on German, 2.91% on Spanish, and 4.27% on Japanese compared to the MLP-based detectors. It should be noted that although incorporating context affected the EER of some AFs more than others, the MLPwC-based detectors outperform the MLP-based detectors for 119 of the 120 monolingual AFs.

3.6.3 Multilingual AF Detection

In the previous section, it was shown that MLPwC-based detectors outperform GMM- and MLP-based AF detectors on the task of monolingual AF detection. Assuming that MLPwC-based detectors will also yield the best performance for the task of multilingual AF detection, one set of MLPwC-based AF detectors was created using all languages. The number of training utterances from each language was chosen such that the multilingual AF detectors were created using approximately the same amount of speech data as the monolingual AF detectors. For example, if an AF existed in all four languages only 25% of the training utterances from each language were used to train the detector. This was done in an effort to minimize the effect of additional training data on de-

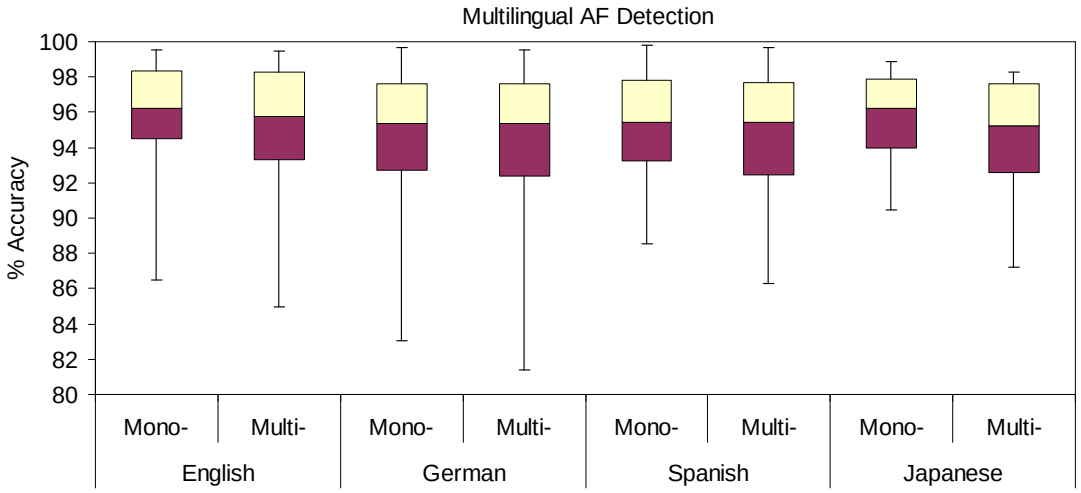


Figure 3.6: Accuracy rates of the monolingual (Mono-) and multilingual (Multi-) MLPwC-based AF detectors on the development partition.

tection performance. As in Section 3.6.2, the detectors were evaluated on the development partition of each language and the reference transcriptions were generated using the procedure discussed in Section 3.3. Note that seven of the AFs only exist in a single language and no multilingual training data was available; these AFs are excluded from the results presented in this section.

Accuracy

The accuracy of the multilingual AF detectors for each language is shown in Figure 3.6. For comparison purposes the results of the monolingual MLPwC-based detectors are also included. The complete results can be found in Appendix A.

From Figure 3.6 we can see that roughly 75% of the AFs in each language are correctly classified at least 92% of the time and approximately 25% of the AFs are correctly classified at least 97% of the time with the multilingual AF detectors. The minimum accuracy is 80.35% for the German AF *alveolar* and the maximum is 99.68% for the Spanish AF *affricate*. Compared to the monolingual AF detectors, the median accuracy decreased by 0.46% on English, 0.04% on German, 0.01% on Spanish, and 0.96% on Japanese. The maximum reduction in accuracy is 4.33% for the Japanese AF *alveolar*. It is interesting to note that an improvement in detection performance was

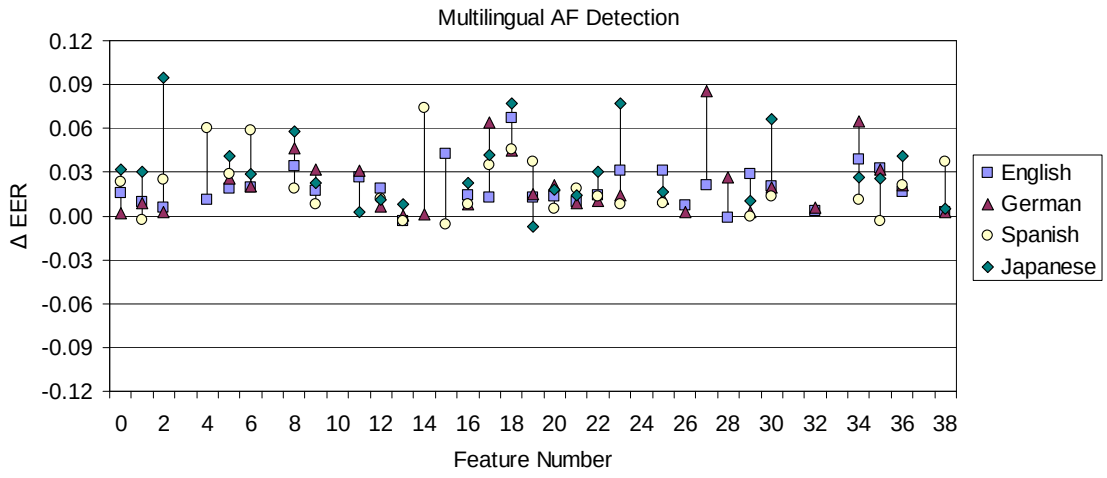


Figure 3.7: Difference in EER between the monolingual and multilingual MLPwC-based AF detectors on the development partition. A positive change in EER represents a decrease in system performance.

obtained with the multilingual detectors for 17 of the 113 AFs scored, where the maximum increase in accuracy is 1.00% for the German AF *open*.

In comparison, the multilingual GMM-based AF detectors created by Stüker [25] decreased the mean accuracy by 3.43% on English, 4.00% on German, 4.74% on Spanish, and 4.28% on Japanese compared to their monolingual counterparts. The mean accuracy of the MLPwC-based AF detectors created in this thesis decreased by 0.69% on English, 0.43% on German, 0.57% on Spanish, and 1.49% on Japanese when using multilingual instead of monolingual detectors. It should be noted that the multilingual detectors created by Stüker were trained using an additional language (Chinese). Also, recall that in [25] the GMM-based AF detectors were trained and evaluated using only speech frames that were time-aligned with the middle states of phoneme models, whereas in this thesis all frames of speech were used.

EER

The difference in EER between the monolingual and multilingual MLPwC-based AF detectors for each language is shown in Figure 3.7. The feature numbers along the horizontal axis correspond to those given in Table 3.4. Note that if the multilingual detector for a given AF decreased the detection performance (*i.e.*, made the EER worse) the change in EER is positive, and negative otherwise.

Overall, the multilingual AF detectors performed worse than the monolingual AF detectors. The median increase in EER for each language is as follows: 1.68% on English, 1.46% on German, 1.35% on Spanish, and 2.67% on Japanese. The maximum increase is 9.51% for the Japanese AF *labiodental* (2) and the maximum reduction is 0.77% for the Japanese AF *affricate* (19). From Figure 3.7 we can see that for most AFs the change in EER varies based on the language; however, for some AFs a similar change in detection performance is obtained for all languages. For example, the change in EER for the AFs *plosive* (12), *nasal* (13), and *unvoiced* (21) is less than 2.0% for all languages. This may indicate that in terms of acoustics, some AFs are more similar across different languages than others.

Chapter 4

Decoding Experiments

The two main decoding tasks that are useful for evaluating the performance of a speech recognizer are word and phoneme recognition. A word recognizer hypothesizes both the phoneme and word sequence of a spoken utterance, whereas a phoneme recognizer hypothesizes only the phoneme sequence. In general, word recognition is performed using either a 2- or 3-gram LM created using the procedure discussed in Section 2.5. Phoneme recognition can be performed by using an N-gram LM trained on phoneme sequences, rather than word sequences.

This chapter discusses how the scores from the AF detectors were used as the feature set for an HMM-based speech recognizer and presents the results obtained. Section 4.1 discusses the performance metrics used to evaluate the performance of the recognizers. Section 4.2 describes the experiments performed using the monolingual AF detectors. Finally, Section 4.3 presents the results obtained with the multilingual AF detectors.

4.1 Performance Metrics

The performance of a speech recognizer can be measured by comparing the hypothesized word or phoneme sequence to a reference transcription. There are three different types of errors that a speech recognizer can produce: deletions, substitutions, and insertions. Deletions are when a phoneme or word is missed, substitutions are when a word or phoneme is incorrectly hypothesized, and insertions are when a phoneme or word is hypothesized that does not exist in the reference transcription.

	the	sun	shined
Reference	ð ə	s ʌ n	ʃ aɪ n d
Hypothesized	ð	s ʌ m	ʃ aɪ n i d
Error	Del	Sub	Ins

Figure 4.1: The three possible types of recognition errors are Deletions (Del), Substitutions (Sub), and Insertions (Ins). This example illustrates each error type by comparing a reference phoneme transcription to the output of a phoneme recognizer (Hypothesized) for the word sequence *the sun shined*.

Figure 4.1 illustrates these different types of errors for a phoneme recognition task. Two metrics that are useful for evaluating the performance of a speech recognizer are *Percent Correct* and *Percent Accuracy*, which can be calculated as follows

$$\text{Percent Correct} = \frac{N - D - S}{N} \times 100\% \tag{4.1}$$

$$\text{Percent Accuracy} = \frac{N - D - S - I}{N} \times 100\% \tag{4.2}$$

where N is the number of reference phonemes, D is the number of deletions, S is the number of substitutions, and I is the number of insertions. Note that the quantity $N - D - S$ is equal to the number of reference phonemes correctly identified.

4.2 Monolingual

This section describes how the scores from the monolingual AF detectors were used as the feature set for an HMM-based speech recognizer and presents the results obtained. First, a vector was formed using the scores from the individual AF detectors. Note that the dimension of this vector is language-dependent and is equal to the number of AFs in the language. The scores for the GMM-based AF detectors were calculated according to Equation 3.6 and the scores for each MLP- and MLPwC-based AF detector were computed by subtracting the output of the absent unit from the output of the present unit. As in Section 3.6.1, the output activation function was removed prior to computing the scores for the MLP- and MLPwC-based detectors so that the scores more closely approximate a Gaussian distribution.

Certain AFs tend to be highly correlated with other AFs. For example, in English most front vowels are rounded and all vowels are normally voiced. In order to use diagonal covariance matrices in the HMMs it is best to decorrelate the individual AF scores. This can be accomplished by using the Karhunen-Loéve Transform (KLT) [31]. The KLT is a linear transform that projects the original data onto a set of orthonormal basis vectors. Define the matrix X as the original data arranged in columns; *i.e.* the number of rows equals the vector dimension and the number of columns equals the number of samples. In addition, define the matrix U as the basis vectors arranged in columns. The KLT is performed as follows

$$Y = U^T X \tag{4.3}$$

where Y is the transformed data. The basis vectors are data dependent and are defined as the eigenvectors of the covariance matrix estimated from X, that is

$$\Sigma u_j = \lambda_j u_j \tag{4.4}$$

where Σ is the covariance matrix estimated from X, u_j is the j^{th} basis vector, and λ_j is the corresponding eigenvalue. In the remainder of this chapter, the feature sets created with the scores from the monolingual GMM-, MLP-, and MLPwC-based AF detectors will simply be referred to as GMM, MLP, and MLPwC feature sets.

A second set of feature vectors that included delta coefficients were also created for each condition. The delta coefficients were calculated using Equation 2.12 and appended prior to performing the KLT. Denote the GMM, MLP, and MLPwC feature sets that include deltas as GMM+D, MLP+D, and MLPwC+D.

4.2.1 Phoneme Recognition Results

Context-independent HMMs were created for each language and feature set using the HTK software package. The procedure discussed in Section 2.4 was used to train the models. Phonemes were modeled using three emitting states, each of which included 32 mixture components with diagonal covariance matrices. In addition, the MFCC-based phoneme models discussed in Section 3.3 were

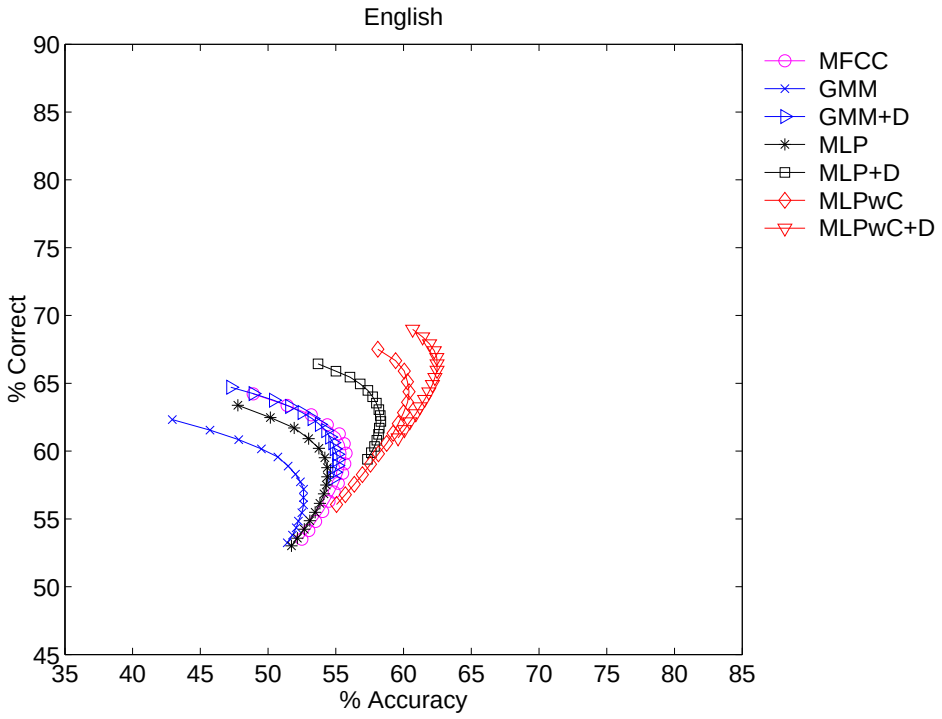
used as a baseline system. It should be emphasized that the MFCC feature set includes energy, delta, and acceleration coefficients.

The recognizers were evaluated on the development set of each language. Recall from Section 2.6.1 that a LM scale factor s and word insertion penalty p are used in the decoder. In order to more directly compare the acoustical models, for the first experiment the phoneme probabilies were ignored by setting $s = 0$ in Equation 2.50; thus only p needed to optimized. Figures 4.2 and 4.3 plot the percent correct percent and accuracy of the recognizers for each language, where each symbol type represents a different feature set and each symbol instance plots the performance for $p = 0, -2, -4, \cdots, -30$. The MFCC system forms the baseline.

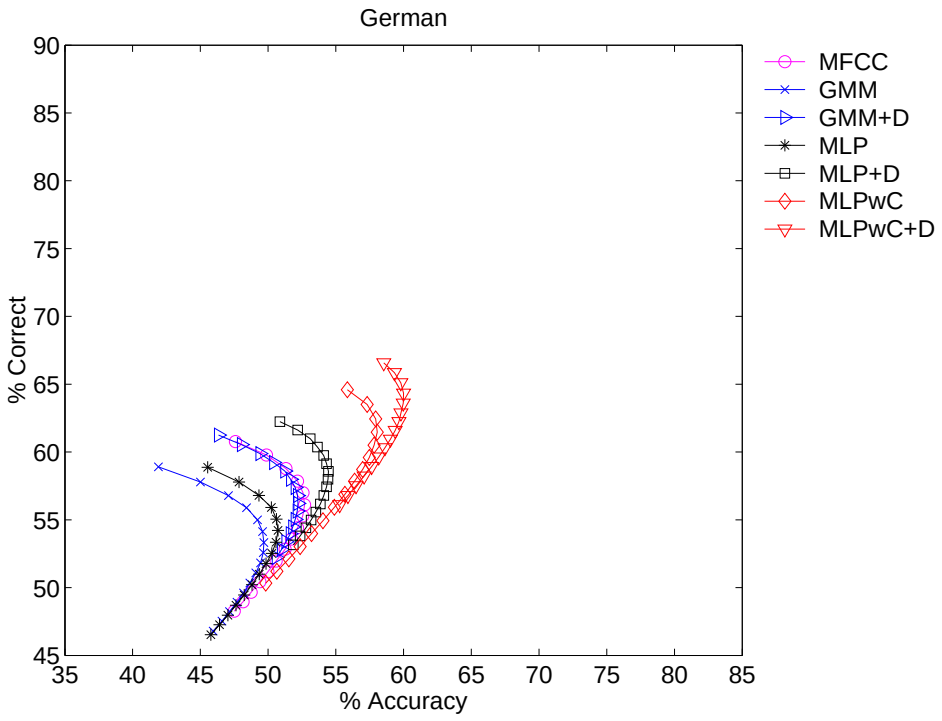
Consider first the AF-based feature sets without delta coefficients. Overall, the GMM features yield the lowest performance. An increase in percent correct and percent accuracy is obtained with the MLP features on all languages except German, where a similar percent correct is obtained for both systems. This improvement is the most substantial on Japanese: if p is chosen to maximize the percent correct for each system an increase of 2.47% correct is obtained with the MLP features, whereas if p is chosen to maximize the percent accuracy the MLP features yield an increase of 3.33% accuracy. Across all languages, however, the performance of the GMM and MLP features is worse than the baseline MFCC system. The MLPwC features outperform the GMM, MLP, and MFCC features sets on all languages. Compared to the MFCC systems, the maximum improvement in system performance is obtained on German: an increase of 3.82% correct and 5.36% accuracy is obtained when p is chosen to maximize these metrics.

Delta coefficients were found to improve all feature sets. On English and German the difference in maximum percent correct and maximum percent accuracy[1] between the GMM+D and MFCC systems is less than 0.65%, whereas on Spanish and Japanese the GMM+D features decrease the maximum percent correct and maximum percent accuracy by more than 1.40%. The MLP+D feature set outperforms the MFCC system on all languages except Japanese, where a similar percent accuracy is obtained for both systems. The best performance is obtained with the MLPwC+D feature set. Compared to the MFCC systems, an increase in maximum percent correct of 4.76% on English, 5.80% on German, 2.96% on Spanish, and 3.13% on Japanese is obtained with the

[1] maximum percent correct is defined here as the percent correct when p is chosen to give the maximum; maximum percent accuracy is defined in a similar manner

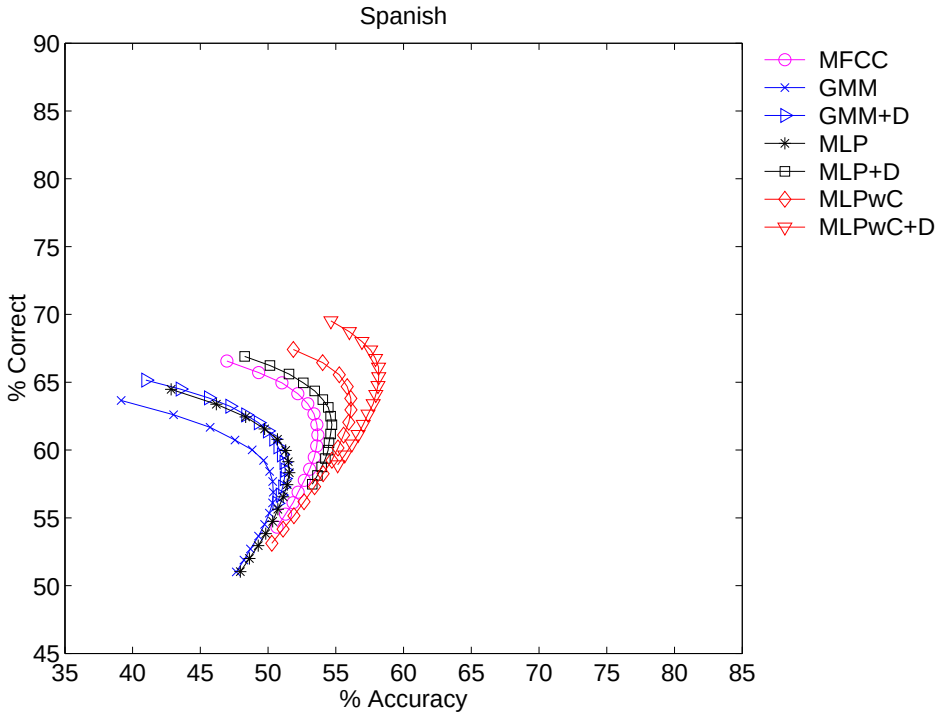


(a)

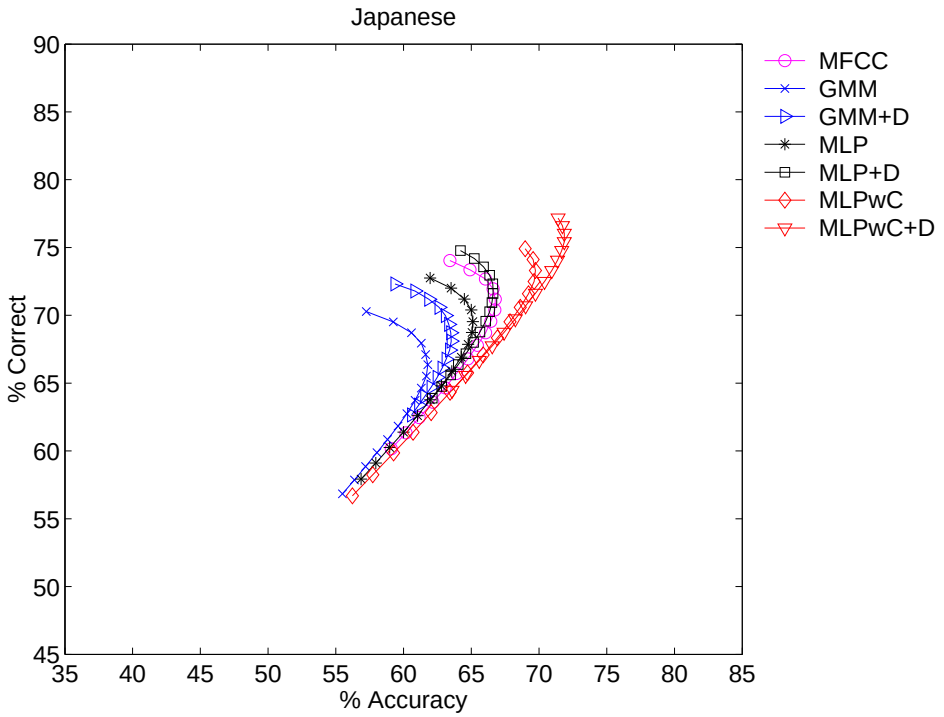


(b)

Figure 4.2: Percent correct and accuracy for (a) English and (b) German on the development partition. Each symbol type represents a different feature set and each symbol instance plots the system performance for a different word insertion penalty p. The MFCC features form the baseline system.



(a)



(b)

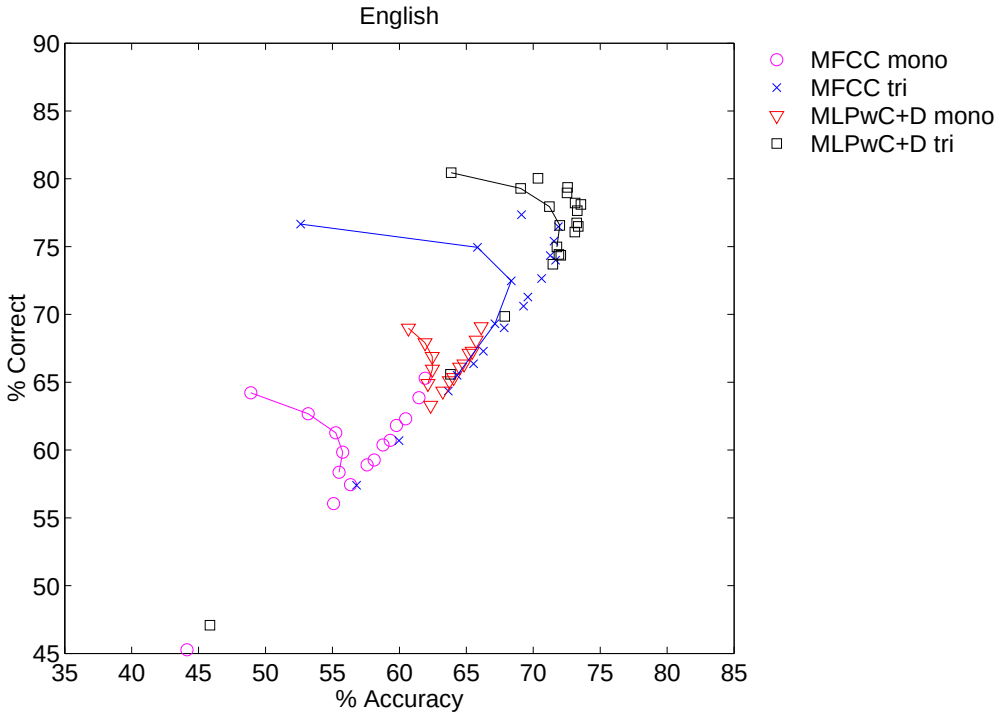
Figure 4.3: Percent correct and accuracy for (a) Spanish and (b) Japanese on the development partition. Each symbol type represents a different feature set and each symbol instance plots the system performance for a different word insertion penalty p. The MFCC features form the baseline system.

MLPwC+D features; the maximum accuracy increased by 6.71% on English, 7.30% on German, 4.49% on Spanish, and 5.10% on Japanese.

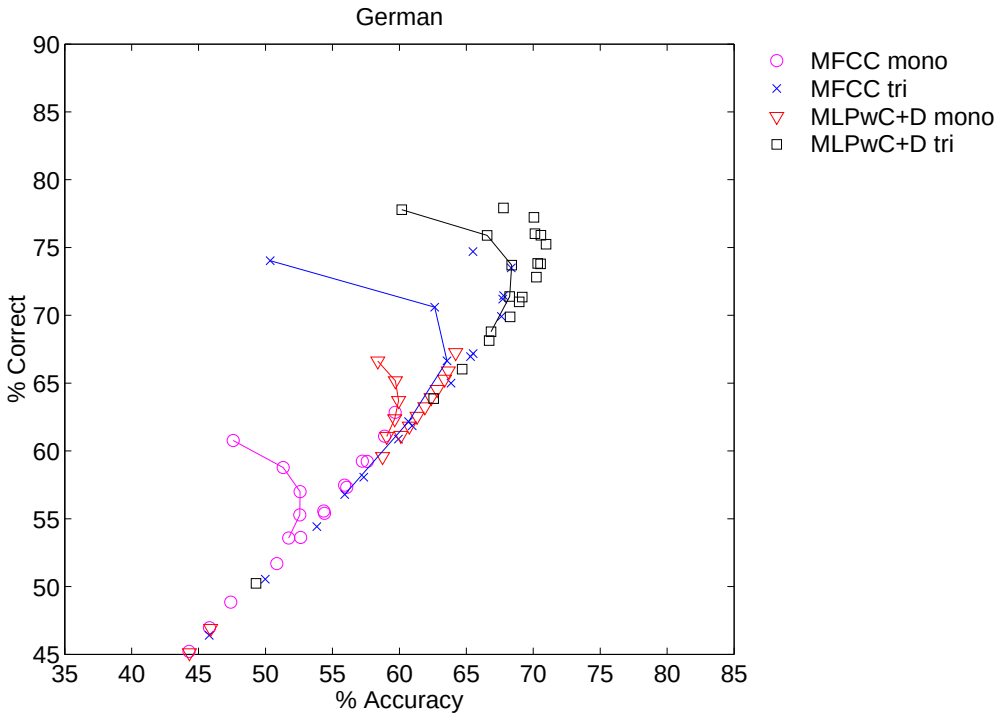
In addition to the previous experiment, triphone models were created for each language using the MFCC and MLPwC+D feature sets. Triphones were modeled using the same HMM topology as monophones, except that each state included only 16 mixture components. The number of mixture components was reduced because modeling context decreases the amount of speech data available to train each model. Recall from Section 2.4 that state clustering it is often neccessary when using context-dependent models, which can be performed using a decision-tree-based method when a set of context questions is supplied. Questions were created based on each AF using the articulatory-to-phoneme mappings. For example, consider the AF *bilabial* which is the place of articulation for the English phonemes /p/, /b/, and /m/. A left contextual question L=bilabial was formed that splits the phonemes of interest into two groups using the following procedure: the phoneme is assigned to group 1 if the preceeding phoneme is /p/, /b/, or /m/, and group 2 otherwise. Similarly, a right contextual question R=bilabial was formed that groups phonemes based on if the following phoneme is /p/, /b/, or /m/. This procedure was repeated to create a right and left contextual question for each AF.

The monophone and triphone MFCC and MLPwC+D systems were also evaluated on the development partition using a bigram phoneme LM. The LM for each language was created using the phoneme transcriptions derived from the training partition. Figures 4.4 and 4.5 plot the percent correct and percent accuracy of the recognizers for each language, where each symbol type represents a different feature set and model type, and each symbol instance plots the performance for a different value of s and p. The monophone and triphone systems were both evaluated for $s = 0, 5, 10, 15$, although it was found that different word insertion penalties were needed to optimize the performance of each system: $p = 0, -4, -8, -12, -16$ were used for the monophone systems and $p = 0, -15, -30, -45, -60$ were used for the triphone systems. Note that the system performance obtained using all combinations of s and p is not shown for some languages.[2] The symbols connected by lines represent the system performance when $s = 0$; that is, the phoneme

[2]High values of s lead to a substantial decrease in system performance for some languages; only the best performance is shown in Figures 4.4 and 4.5

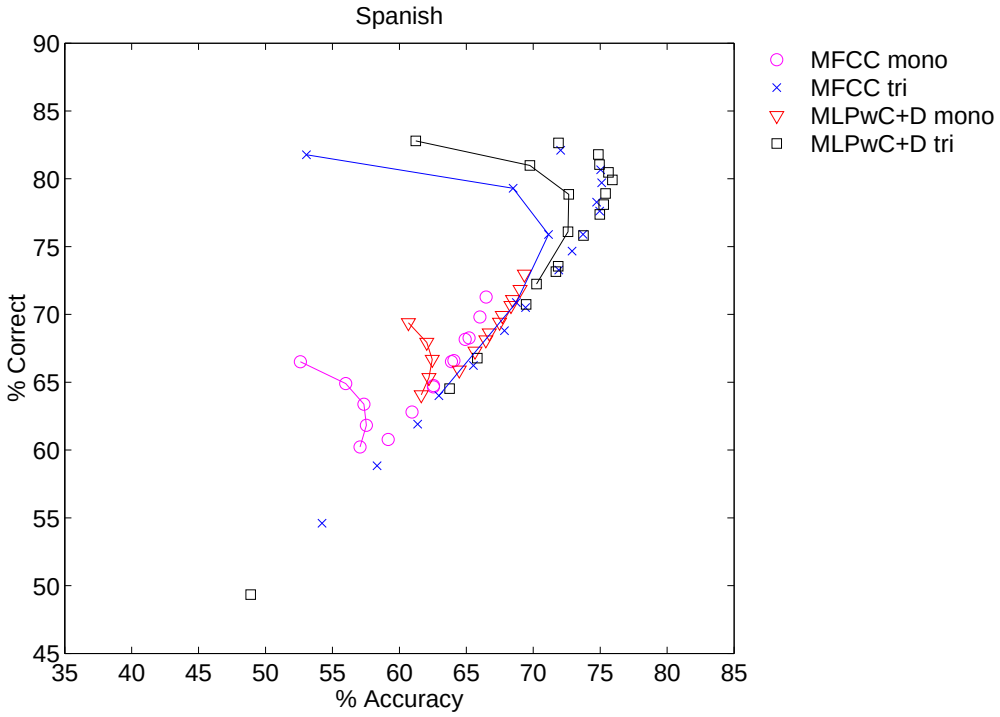


(a)

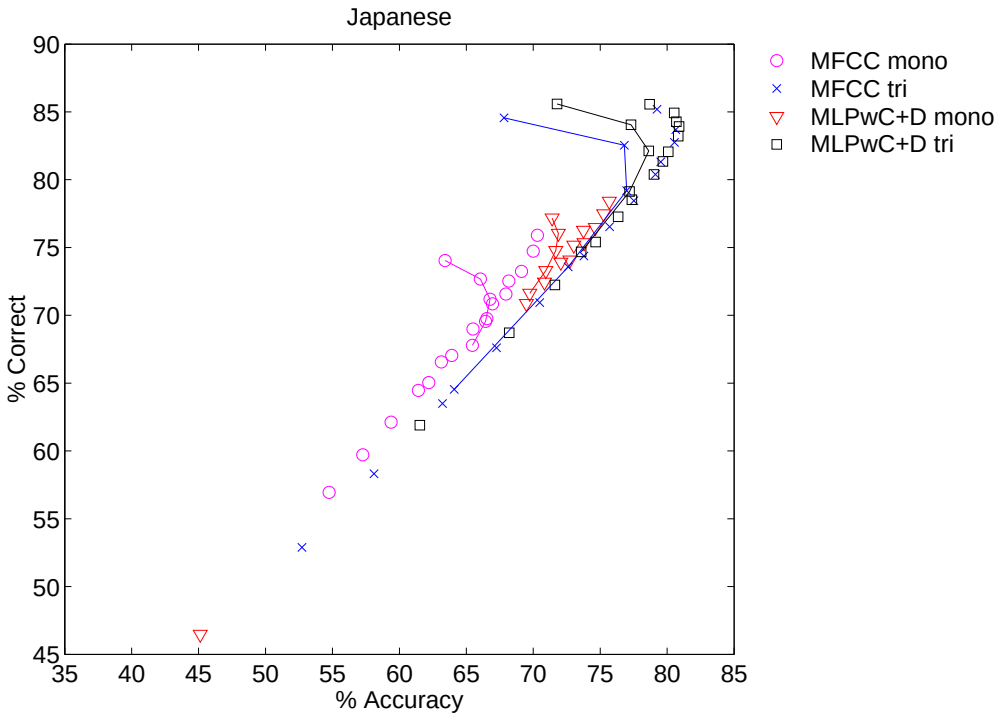


(b)

Figure 4.4: Percent correct and accuracy for (a) English and (b) German on the development partition. Each symbol type represents a different feature set with either monophone (mono) or triphone (tri) models, and each symbol instance plots the system performance for a different LM scale s and word insertion penalty p. The MFCC features form the baseline system.



(a)



(b)

Figure 4.5: Percent correct and accuracy for (a) Spanish and (b) Japanese on the development partition. Each symbol type represents a different feature set with either monophone (mono) or triphone (tri) models, and each symbol instance plots the system performance for a different LM scale s and word insertion penalty p. The MFCC features form the baseline system.

probabilities are ignored when computing Equation 2.50.[3] The MFCC features form the baseline system.

Consider first the monophone systems. Including phoneme probabilities in the decoder increases the percent correct and accuracy of all systems. Furthermore, the MLPwC+D features outperform the MFCC features on each language. In terms of maximum percent correct the most substantial increase is 4.39% on German, whereas the greatest increase in maximum percent accuracy is 5.37% on Japanese. It is interesting to note that on English and German the MLPwC+D systems decoded with $s = 0$ yield a higher maximum percent correct than the MFCC systems decoded with $s = 0, 5, 10, 15$, and the difference in maximum percent accuracy between these systems is less than 0.55%. Thus, on English and German the MLPwC+D systems decoded without a LM yield comparable performance to a MFCC system decoded with a LM. On Japanese, the MLPwC+D system decoded with $s = 0$ yields an increase in maximum percent correct of 1.27% and an increase in maximum precent accuracy of 1.56% compared to the MFCC system.

Modeling triphones instead of monophones leads to an improvement in system performance for both feature sets. On English and German the MLPwC+D system outperforms the MFCC feature set, whereas similar performance is obtained with both feature sets on Spanish and Japanese. Compared to the MFCC systems, an increase in maximum percent correct of 2.69% on English and 3.22% on German is obtained with the MLPwC+D features; the maximum accuracy increased by 1.69% on English and 2.60% on German. The difference in maximum percent correct and accuracy between the MLPwC+D and MFCC systems is less than 0.80% on Spanish and Japanese. Note that on English and German the MLPwC+D systems decoded with $s = 0$ yields a higher maximum percent correct than the MFCC systems and a difference in maximum accuracy of less than 0.11%.

In this section it was shown that MLPwC features outperform GMM and MLP features on a phoneme recognition task when using monophone models and ignoring phoneme probabilities in the decoder. Delta coefficients were found to improve the performance of each feature set, and the best overall recognition performance was obtained using MLPwC+D features. Compared to the MFCC systems, the MLPwC+D features increased the maximum percent correct by up to 5.80% and the maximum percent correct accuracy by up to 7.30%. When a bigram phoneme LM was used,

[3] Note that this is the same as was done in the experiments shown in Figures 4.4 and 4.5.

the MLPwC+D features outperformed the MFCC features, increasing the maximum percent correct by up to 4.39% and the maximum percent accuracy by up to 5.37%. Triphone models were also created using the MLPwC+D and MFCC feature sets and evaluated on a phoneme recognition task. On English and German the MLPwC+D features yielded an improvement over the MFCC system, whereas similar performance was obtained on Spanish and Japanese.

4.2.2 Word Recognition Results

The monophone and triphone MFCC and MLPwC+D systems were also evaluated on a word recognition task. A bigram LM was created for each language using the word transcriptions from the training partition. It should be noted that a relatively small amount of text was available for training the LMs. Each system was evaluated using all combinations of s and p as described in the previous section, except that $s = 0$ was ignored. Figure 4.6 shows the percent correct and percent accuracy of each system for the values of s and p that yield the highest average percent correct and accuracy.

Consider first the monophone systems. Compared to the MFCC systems, the MLPwC+D features decrease the percent correct and percent accuracy on all languages except German, where an increase of 2.07% correct and 2.49% accuracy is obtained. The most substantial decrease in system performance is obtained on English: the percent correct decreased by 1.37% and the percent accuracy decreased by 2.69%.

Modeling triphones instead of monophones increases the performance of both feature sets on all languages except English, where a decrease in accuracy of 1.10% is obtained with the MLPwC+D features. The triphone models, however, yield a greater increase in percent correct and percent accuracy for the MFCC features than the MLPwC+D features. Compared to the MFCC systems, the MLPwC+D features decrease the system performance for all languages. The maximum decrease in system performance obtained with the MLPwC+D features is 7.02% correct and 10.71% accuracy on English; the minimum decrease in system performance is 1.67% correct and 3.92% accuracy on Japanese.

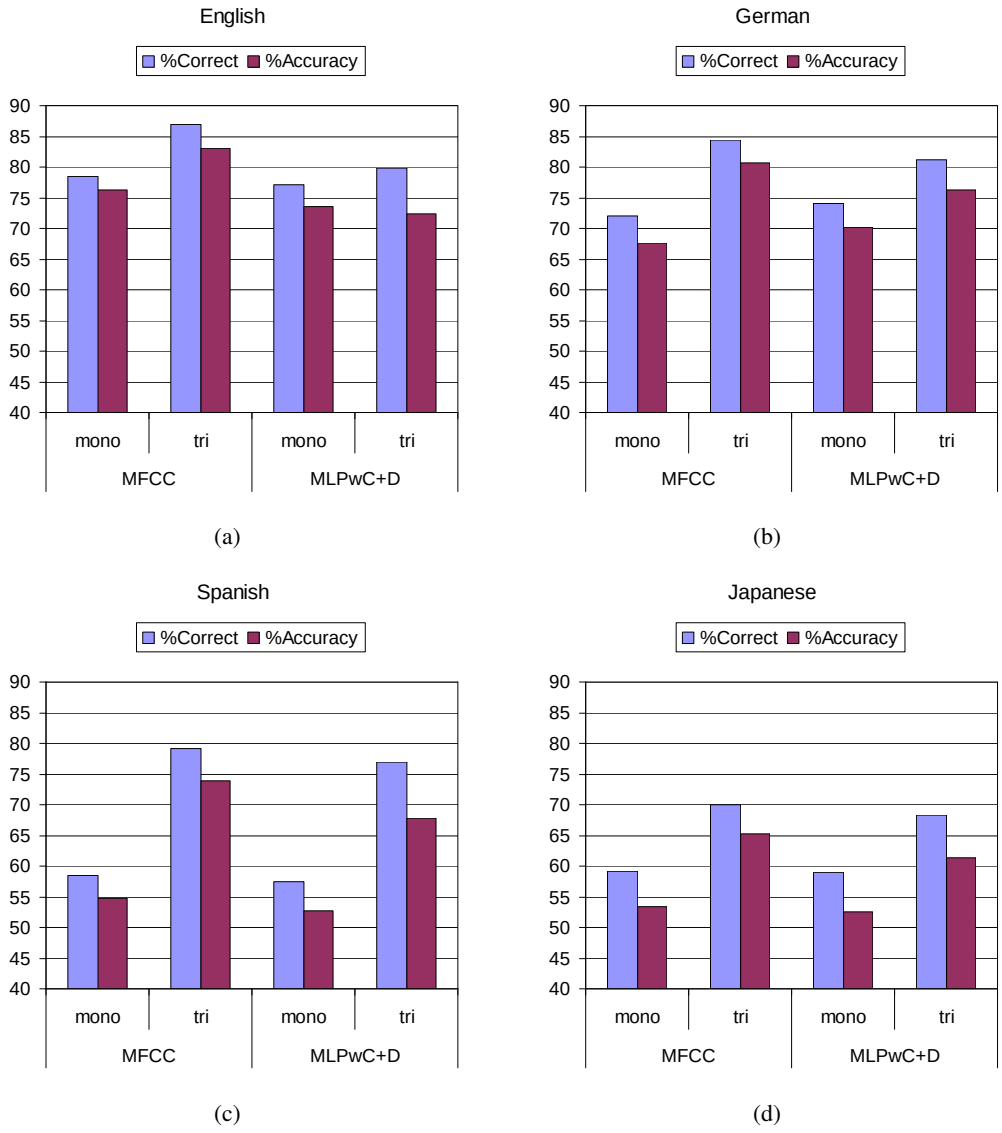


Figure 4.6: WER on the development partition obtained using the MFCC and MLPwC+D feature sets with monophone (mono) and triphone (tri) models. The system performance is shown for the values of s and p that yield the highest average percent correct and accuracy.

4.3 Multilingual

This section describes how the scores from the multilingual AF detectors were used as the feature set for an HMM-based speech recognizer and presents the recognition results obtained. Recall from Section 4.2 that the dimension of the feature vectors derived from the monolingual AF detectors is language-dependent. One potential disadvantage of this method is that it is not possible to train phoneme models using speech data from multiple languages. Although multilingual phoneme mod-

els were not created in this thesis, a language-independent[4] feature vector was nonetheless formed by using the scores from all of the multilingual MLPwC-based AF detectors. The procedure discussed in Section 4.2 was used to calculate the scores for each AF.

In the previous section it was shown that including delta coefficients improves the performance of the feature sets derived from the monolingual AF detectors. Assuming that a similar trend in system performance will be observed with the multilingual AF detectors, delta coefficients were also included. The delta coefficients were calculated according to Equation 2.12, and the feature vectors were processed with a KLT. In the remainder of this chapter, this feature set will be referred to as MLPwC+D ML.

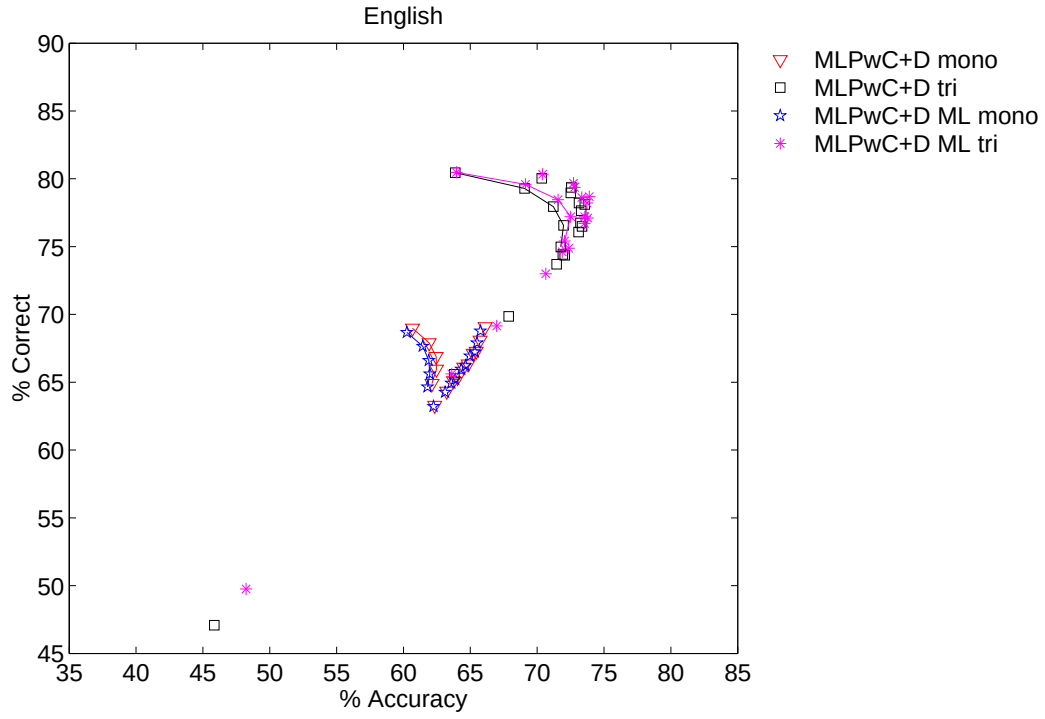
4.3.1 Phoneme Recognition Results

Monophone and triphone models were created for each language and evaluated on the development partition using the same procedure as discussed in Section 4.2. It should be emphasized that the only difference between the recognizers created in this section and those described in Section 4.2 is the feature set used. Figures 4.7 and 4.8 plot the percent correct and percent accuracy of the MLPwC+D ML recognizers for each language; for comparison purposes, the results of the MLPwC+D system are also included.

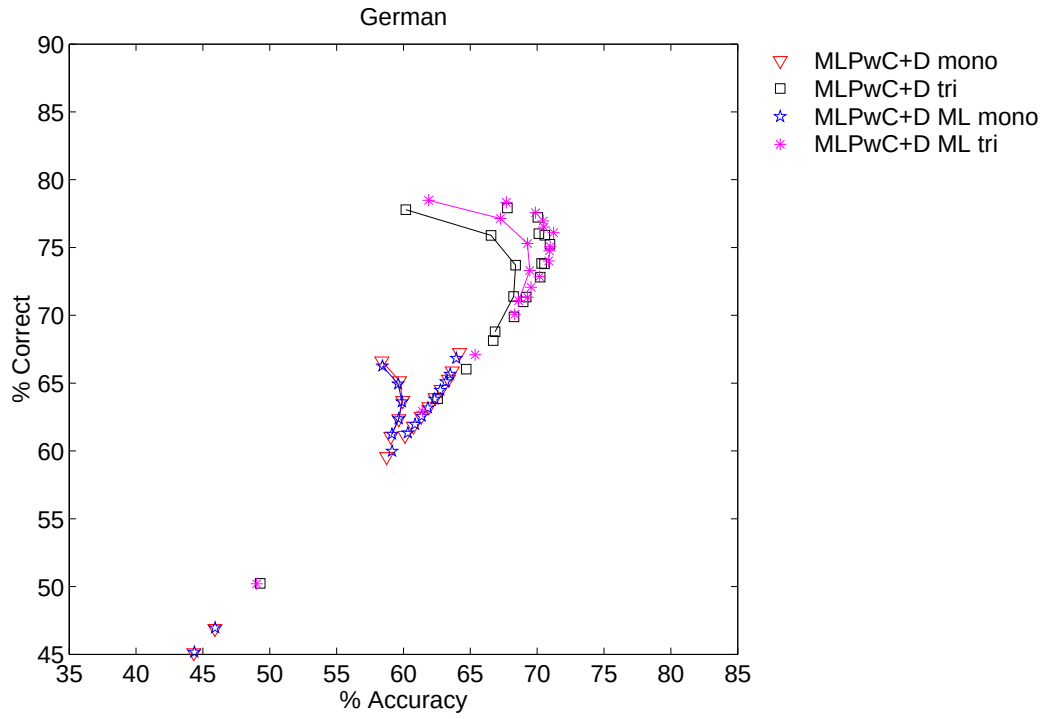
Consider first the monophone systems. On English, German, and Spanish, the difference in maximum percent correct and accuracy between the MLPwC+D and MLPwC+D ML systems is less than 0.41%. On Japanese, the MLPwC+D ML features yield a decrease in maximum percent correct of 1.44% and a decrease in maximum percent accuracy of 1.72% compared to the MLPwC+D features; however, the MLPwC+D ML features increase the maximum percent correct by 1.07% and the maximum accuracy by 3.65% compared to the MFCC system. Note that in Section 3.6.3 it was observed that the multilingual MLPwC-based AF detectors lead to the largest overall decrease in accuracy and EER on Japanese.

With triphone models, the difference in maximum percent correct and accuracy between the MLPwC+D and MLPwC+D ML features is less than 0.75% for all languages. Note that when the

[4] the term *language-independent* is used here to refer only to the languages considered in this thesis

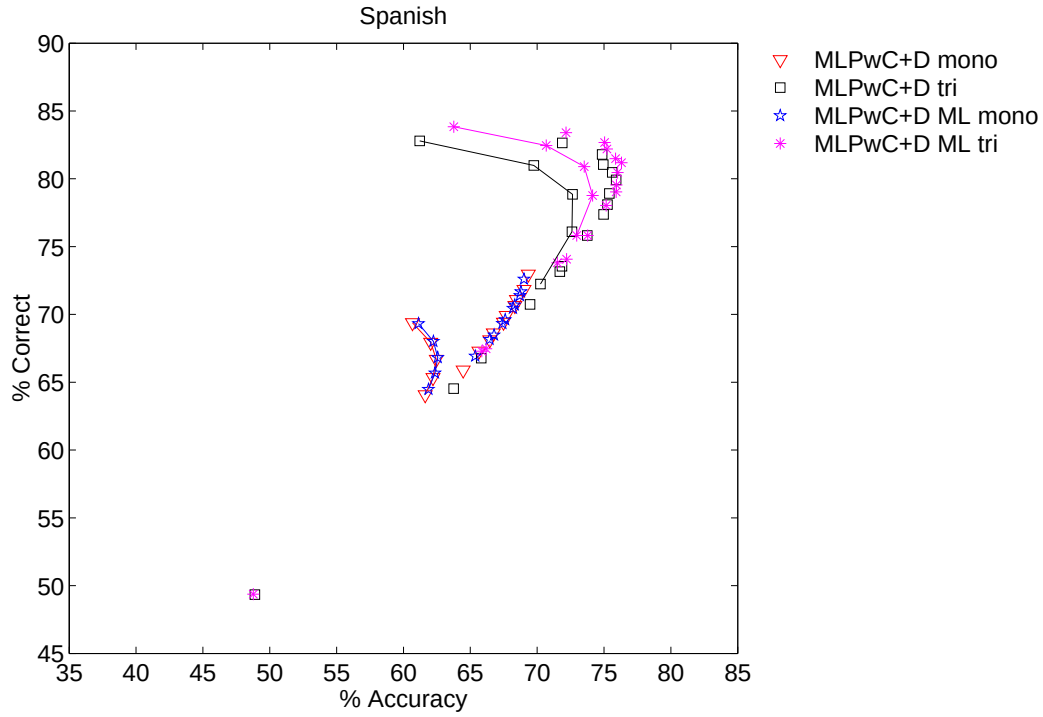


(a)

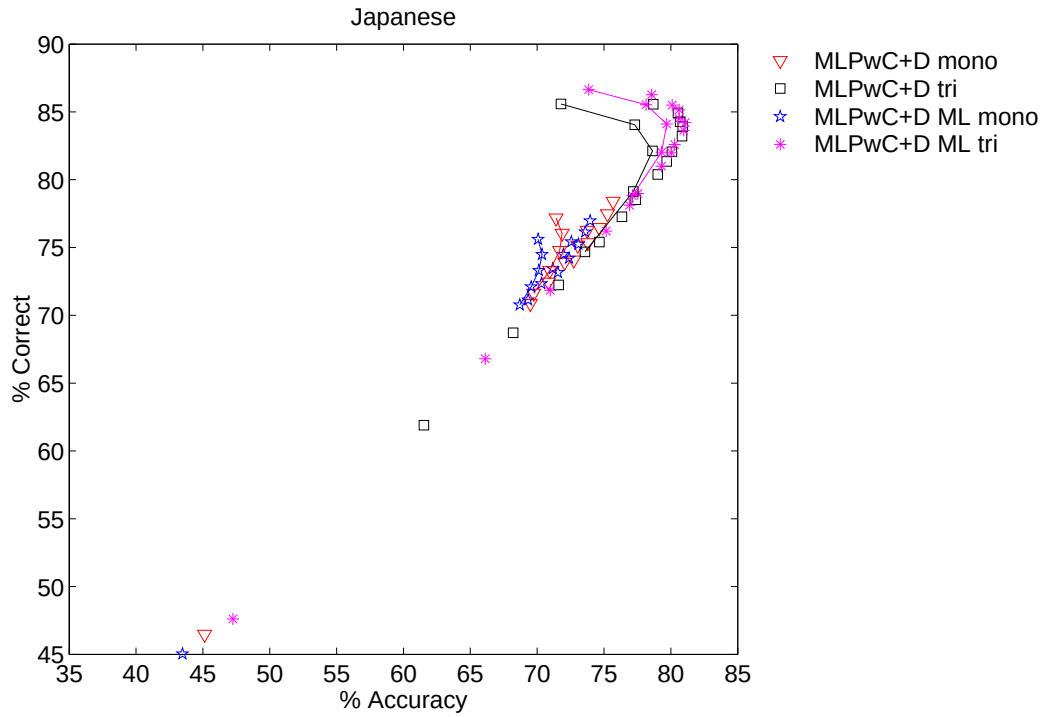


(b)

Figure 4.7: Percent correct and accuracy for (a) English and (b) German on the development partition. Each symbol type represents a different feature set with either monophone (mono) or triphone (tri) models, and each symbol instance plots the system performance for a different LM scale s and word insertion penalty p.



(a)



(b)

Figure 4.8: Percent correct and accuracy for (a) Spanish and (b) Japanese on the development partition. Each symbol type represents a different feature set with either monophone (mono) or triphone (tri) models, and each symbol instance plots the system performance for a different LM scale s and word insertion penalty p.

phoneme probilities are ignored the difference in maximum percent correct and accuracy between the MLPwC+D and MLPwC+D ML features is less than 0.53% on English. On German, Spanish, and Japanese, the MLPwC+D ML features outperform the MLPwC+D features when the phoneme probabilites are ignored; however, the improvement in maximum precent correct and accuracy is less than 1.50%.

4.3.2 Word Recognition Results

The monophone and triphone MLPwC+D ML systems were also evaluated on a word recognition task. Each system was evaluated for all combinations of s and p described in section 4.2.2 and the same bigram LM was used. Figure 4.9 shows the percent correct and percent accuracy of each system for the values of s and p that yield the highest average percent correct and accuracy; for comparison purposes, the results of the MLPwC+D system are also included.

Consider first the monophone systems. The MLPwC+D ML features yield worse performance than the MLPwC+D systems for all languages. The maximum decrease in percent correct is 2.22% on Japanese and the maximum decrease in percent accuracy is 3.18% on German. It should be noted that the difference in percent correct and percent accuracy between the MLPwC+D and MLPwC+D ML feature sets on English is less than 0.33%.

Modeling triphones instead of monophones increases the performance of the MLPwC+D ML feature set on all languages except English, where a decrease in accuracy of 1.85% is obtained. Compared to the MLPwC+D system, the MLPwC+D ML features yield a decrease in percent correct and percent accuracy for all languages. The maximum decrease in percent correct is 3.75% on Spanish and the maximum decrease in percent accuracy is 6.01% on Japanese. On English, the difference in percent correct and percent accuracy between the MLPwC+D and MLPwC+D ML feature set is less than 0.78%.

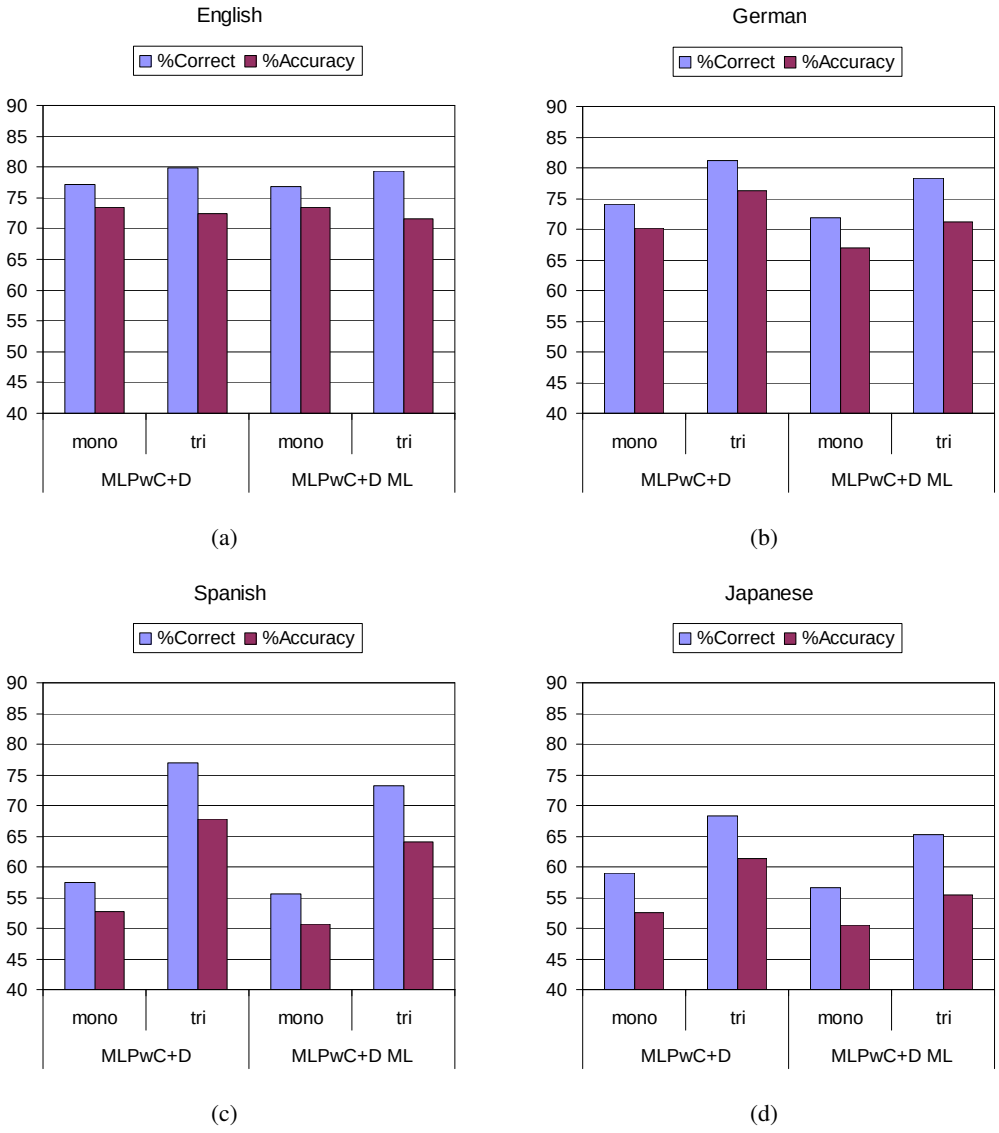


Figure 4.9: WER on the development partition obtained using the MLPwC+D and MLPwC+D ML feature sets with monophone (mono) and triphone (tri) models. The system performance is shown for the values of s and p that yield the highest average percent correct and accuracy.

Chapter 5

English AF Score Fusion

Score fusion is a technique that combines the outputs of multiple classifiers to produce a system with an error rate lower than that of the individual systems. This chapter discusses two fusion experiments that were performed to investigate whether it might be beneficial to fuse the scores from the GMM- and MLP-based AF detectors. As opposed to the previous experiments discussed in this thesis, only English speech is considered in this chapter. Furthermore, the English GMM- and MLP-based AF detectors were evaluated on a different corpus.

This chapter is organized as follows. First, Section 5.1 describes the corpus that was used for the experiments discussed in this chapter. Section 5.2 describes two different procedures that were used to fuse the scores from the GMM- and MLP-based AF detectors. Section 5.3 presents the AF detection results obtained. Finally, Section 5.4 describes how the scores from the AF detectors were used as the feature set for a phone recognizer and presents the recognition results obtained.

5.1 TIMIT Corpus

TIMIT [35] is an English speech database that includes speakers from eight different dialect regions of the United States: New England, Northern, North Midland, South Midland, Southern, New York City, Western, and Army Brat (moved around). This corpus consists of read text selected from phonetically balanced sentences. Speech data were recorded at a 16 kHz sampling rate and 16 bit

Table 5.1: Overview of the training and test sets for the TIMIT corpus. Spkrs is the number of different speakers and M/F gives the distribution of male and female speakers.

Training			Test		
Hours	Spkrs	M/F	Hours	Spkrs	M/F
3.15	462	326/136	2.25	168	112/56

resolution. TIMIT includes a total of 6300 utterances ($\sim$5.4 hours) spoken by 630 different speakers. In contrast to the WSJ1-CSR and GlobalPhone corpora, all speech data are hand-transcribed at the phone level.

The training and complete test partition of the TIMIT corpus were used for all experiments discussed in this chapter. These partitions are pre-defined in the corpus documentation and exclude two utterance spoken by each speaker (1200 utterances are excluded). The database statistics are shown in Table 5.1 and the AF-to-phone mappings used in this thesis are shown in Table 5.2.

5.2 Fusion Procedure

This section describes the procedures used for fusing the scores from the GMM- and MLP-based AF detectors. Both fusion methods discussed in this section used a single MLP for each AF with a form similar to the that described in Section 3.5. All networks included 100 hidden units that used the logistic function as the activation function, and the softmax function was used as the output activation function. In addition, all fusion MLPs included a context window of nine.

In order to obtain the inputs for the fusion MLPs, the GMM- and MLP-based AF detectors created using the WSJ1-CSR corpus were first evaluated on the training partition of the TIMIT corpus. The scores for each GMM-based AF detector were calculated according to Equation 3.6. The scores for the MLP-based AF detectors were calculated by subtracting the output of the absent unit from the output of the present unit. As in Section 3.6.1, the output activation function was removed prior to computing the scores for the MLP-based AF detectors.

The first fusion method simply combines the scores from the GMM- and MLP-based AF detector for each AF. For example, the fusion MLP for the AF *plosive* uses input features consisting

Table 5.2: AF-to-phone mappings for the TIMIT corpus

AF	Phones
CONSONANT	b,bcl,d,dcl,g,gcl,p,pcl,t,tck,k,kcl,dx,q,ch,tcl,jh,s,sh,z,zh,f,th,v,dh, m,n,ng,em,en,eng,l,el,r,w,y,hh,hv,nx
bilabial	b,bcl,p,pcl,m,em
labiodental	f,v
labialvelar	w
dental	th,dh
alveolar	d,dcl,t,tck,dx,s,z,n,en,l,el,r,nx
postalveolar	ch,tcl,jh,dcl,sh,zh
retroflex	er,axr
palatal	y
velar	g,gcl,k,kcl,ng,eng
glottal	q,hh,hv
plosive	b,bcl,d,dcl,g,gcl,p,pcl,t,tck,k,kcl,q
nasal	m,n,ng,em,en,eng,nx
tap,or,flap	dx,nx
fricative	s,sh,z,zh,f,th,v,dh,hh,hv
approximant	r,w,y
lateral,approximant	l,el
affricate	ch,tcl,jh
voiced	b,bcl,d,dcl,g,gcl,dx,jh,z,zh,v,dh,m,n,ng,em,en,eng,l,el,r,w,y,hv,iy, ih,eh,ey,ae,aa,ah,ao,ow,uh,uw,ux,er,ax,ix,axr,nx
unvoiced	p,pcl,t,tck,k,kcl,q,ch,tcl,s,sh,f,th,hh,ax-h
VOWEL	iy,ih,eh,ey,ae,aa,ah,ao,ow,uh,uw,ux,er,ax,ix,axr,ax-h
close	iy,uw,ux,ix
near-close	ih,uh
close-mid	ey,ow
mid	ax,axr,ax-h
open-mid	eh,ah,ao,er
near-open	ae
open	aa
front	iy,eh,ey,ae
near-front	ih
central	er,ax,ix,axr,ax-h
near-back	uh
back	aa,ah,ao,ow,uw,ux
rounded	ao,ow,uh,uw,ux
unrounded	iy,ih,eh,ey,ae,aa,ah,ix
silence	pau,h#,epi

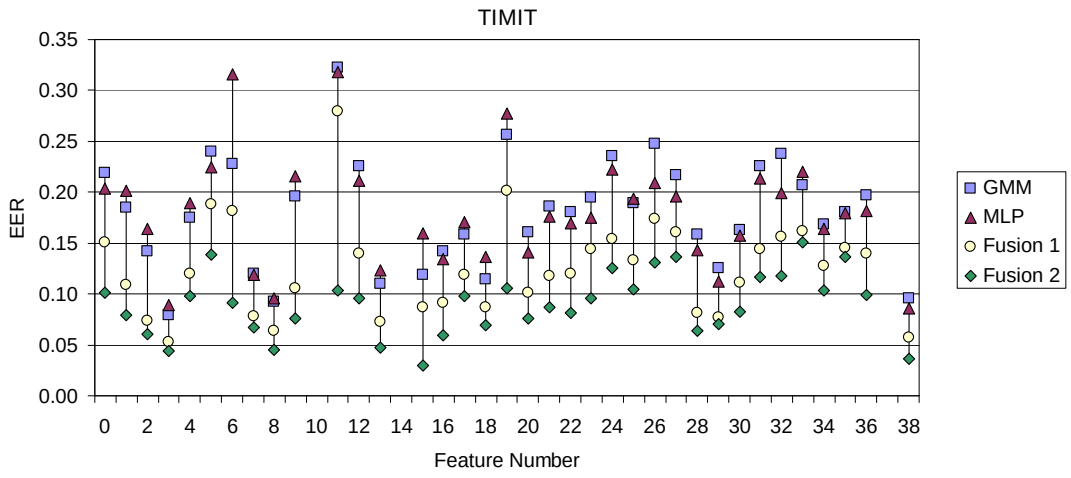


Figure 5.1: EERs of the AF detectors on the TIMIT test partition. The The feature numbers correspond to those given in Table 1.

of the output of the GMM-based plosive detector and the MLP-based plosive detector. This method is referred to as *Fusion 1* in the remainder of this thesis.

The second method creates MLPs for each AF that fuse the scores from all of the GMM- and MLP-based AF detectors; thus the fusion MLP for each AF is provided information about all English AFs from two different classifiers. This procedure is expected to outperform Fusion 1 because of the correlations between individual AFs. In English, for example, all nasals are normally voiced; thus if a speech segment is unvoiced it is unlikely to be a nasal. Denote this procedure as *Fusion 2*.

5.3 AF Detection

The fusion MLPs described in the previous section were evaluated on the test partition of the timit corpus. It should be noted that the reference phone transcriptions were hand-labeled and thus should provide accurate results. The EERs of the individual AF detector created using the Fusion 1 and Fusion 2 procedure are shown in Figure 5.1; for comparison purposes, the results obtained with the GMM- and MLP-based AF detectors are also included. The feature numbers along the horizontal axis correspond to those given in Table 3.4.

The median EER of the GMM-based AF detectors is 18.29%; the minimum EER is 7.95% for the AF *labialvelar* (3), and the maximum is 32.30% for the AF *glottal* (11). The MLP-based AF detectors yield a median EER of 17.81%; the minimum EER is 8.54% for the AF *silence* (38), and the maximum is 31.85% for the AF *glottal* (11).

Fusion 1 improves the detection of all AFs. The median decrease in EER is 4.23% compared to the best GMM- or MLP-based detectors for each AF; that is, the detector with the minimum EER for each AF. The maximum reduction in EER is 9.05% for the AF *velar* (9) and the minimum is 2.75% for the AF *lateral-approximant* (18).

The median EER obtained with Fusion 2 is 9.36%; the minimum EER is 3.01% for the AF *tap or flap* (15) and the maximum is 15.03% for the AF *near-back* (33). The Fusion 2 procedure outperforms the Fusion 1 method for all AFs. Compared to the best GMM- or MLP-based detector, the median decrease in EER obtained with Fusion 2 is 7.93%. The maximum reduction in EER is 21.54% for the AF *glottal* (11) and the minimum is 3.58% for the AF *labialvelar* (3).

5.4 Phone Recognition

This section discusses the phone recognition experiments conducted. The same procedure as described in Section 4.2 was used to form a feature set using the outputs from the GMM- and MLP-based AF detectors. Feature sets were also created from the Fusion 1 and Fusion 2 MLPs by forming a vector using the scores from the individual fusion MLPs and performing a KLT. All feature sets included delta coefficients. In the remainder of this chapter, the feature sets created using the scores from the GMM- and MLP-based AF detectors are simply referred to as GMM and MLP features; similarly, the feature sets formed using the scores from the Fusion 1 and Fusion 2 MLPs are referred to as Fusion 1 and Fusion 2 features. In addition, the MFCC feature set described in Section 3.3 was used to form a baseline system. It should be emphasized that the MFCC feature set includes energy, delta, and acceleration coefficients.

Context-independent and context-dependent HMMs were created for each feature set. Phones were modeled using three emitting states, each of which included 16 mixture components with

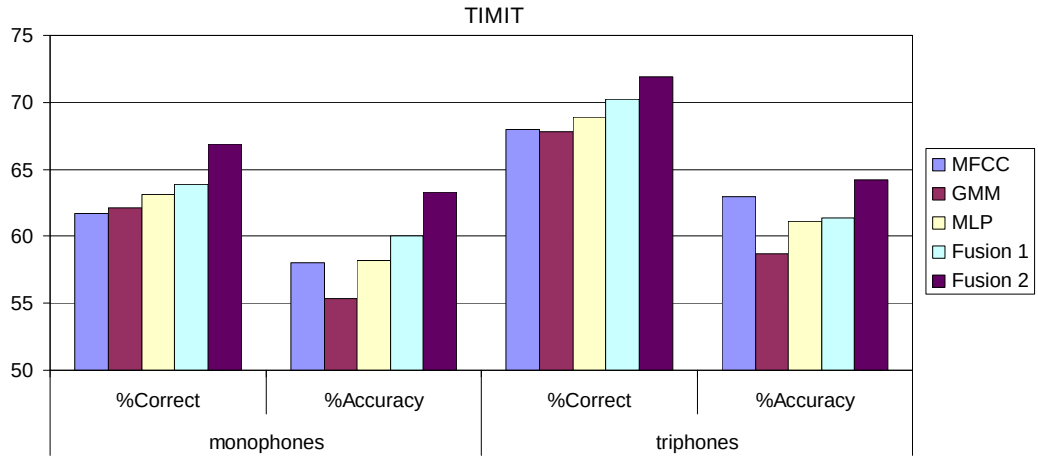


Figure 5.2: Phone recognition results obtained on the TIMIT database using an equally-likely LM.

diagonal covariance matrices. State clustering of the triphone models was performed using a set of decision tree questions created from the articulatory-to-phone mappings shown in Table 5.2.

The monophone and triphone systems were first evaluated using an equally-likely phone LM; that is, the probability of any phone occuring is the same. In contrast to the experiments discussed in Chapter 4, each system was only decoded for $s=1$ and $p=-10$. The percent correct and percent accuracy of each system is shown in Figure 5.2. The MFCC system forms the baseline.

Consider first the monophone systems. The MFCC features yield the lowest percent correct and the GMM features yield the lowest percent accuracy. An increase in percent correct compared to the MFCC and GMM systems is obtained with the MLP features, although the difference in percent accuracy between the MLP and MFCC systems is only 0.13%. The Fusion 1 features outperform the MFCC, GMM, and MLP feature sets: an increase of 2.21% correct and 2.02% accuracy is obtained compared to the MFCC system. The best overall performance is obtained with the Fusion 2 feature set. Compared to the MFCC system, the Fusion 2 features increase the percent correct by 5.19% and the percent accuracy by 5.28%.

Modeling triphones instead of monophones leads to an improvement in system performance for all feature sets. The GMM features yield the worst performance, although the difference in percent correct between the MFCC and GMM feature sets is only 0.17%. Compared to the MFCC system, an increase of 0.99% correct and a decrease of 1.90% accuracy is obtained with the MLP features. The Fusion 1 feature set yields an increase in percent correct of 2.26% and a decrease in

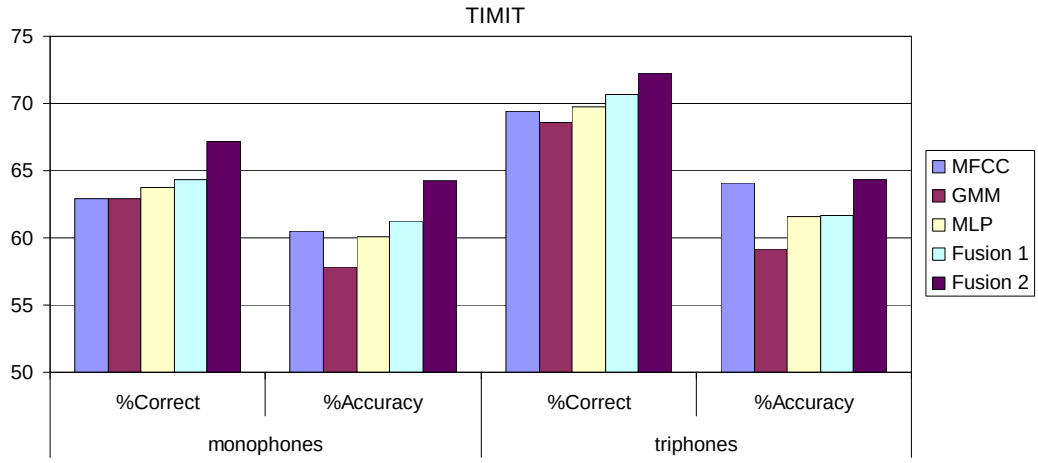


Figure 5.3: Phone recognition results obtained on the TIMIT database using a bigram LM.

percent accuracy of 1.58% compared to the MFCC system. As with the monophone systems, the best overall performance is obtained with the Fusion 2 feature set. Compared to the MFCC system, the Fusion 2 features yield an increase of 3.97% correct and 1.25% accuracy. It is interesting to note that the difference in percent accuracy between the MFCC triphone system and Fusion 2 monophone system is only 0.34%.

The monophone and triphone systems were also evaluated using a bigram phone LM. The LM was created using the phone transcriptions from the training partition. Each system was evaluated for $s = 15$ and $p = -10$. The percent correct and percent accuracy of each system is shown in Figure 5.3. The MFCC system forms the baseline.

Overall, the systems decoded using the bigram LM yield similar trends in performance to those obtained using the equally-likely LM. Using monophone models, the Fusion 2 features yield the best performance. Compared to the MFCC system, an increase of 4.26% correct and 3.78% accuracy is obtained with the Fusion 2 feature set. With triphone models, the Fusion 2 feature set yields an increase in percent correct of 2.84% compared to the MFCC system, although the difference in percent accuracy between the MFCC and Fusion 2 features is only 0.28%. Note that the difference in percent accuracy between the MFCC triphone system and the Fusion 2 monophone system is only 0.13%.

Chapter 6

Conclusion

This thesis considered the task of detecting the presence of AFs in a spoken utterance. MFCCs were extracted from the acoustic speech signal and used as the input to detectors trained to recognize individual AFs. GMM-, MLP-, and MLPwC-based AF detectors were created for English, German, Spanish, and Japanese. In terms of accuracy and EER, the best overall performance was obtained with the MLPwC-based AF detectors. The median accuracy for each language was greater than 95.39% and the median EER was less than 10.62%. Multilingual MLPwC-based AF detectors were also created by using speech data from all four languages. Compared to the monolingual MLPwC-based AF detectors, the median accuracy for each language decreased by less than 0.96% and the median increase in EER for each language was less than 2.67%.

The outputs of the AF detectors were used to form the feature set for a HMM-based phoneme recognizer. Feature vectors with and without delta coefficients were created for each set of monolingual detectors. As an initial experiment, monophone models were used and phoneme probabilities were ignored in the decoder; an MFCC system was used as the baseline. The MLPwC+D feature set yielded the best performance, increasing the maximum percent correct by up to 5.80% and the maximum percent accuracy by up to 7.30%. Furthermore, the MLPwC+D features outperformed the MFCC system when phoneme probabilities were included in the decoder, increasing the maximum percent correct by up to 4.39% and the maximum percent accuracy by up to 5.37%. Triphone models were also created using the MLPwC+D and MFCC feature sets. On English and German, an improvement over the MFCC system was obtained with the MLPwC+D features, whereas simi-

lar performance was obtained on Spanish and Japanese. On a word recognition task, however, the MFCC system outperformed the MLPwC+D system.

The scores from the all of the mulitlingual MLPwC-based AF detectors were also used to form a feature set. Monophone and triphone systems were created for each language. On a phoneme recognition task, the difference in maximum percent correct and maximum percent accuracy between the MLPwC+D and MLPwC+D ML monophone systems was less than 0.41% on all languages except Japanese. The difference in maximum percent correct and maximum percent accuracy between the MLPwC+D and MLPwC+D ML triphone systems was less than 0.75% on all languages. On a word recognition task, the MLPwC+D features outperformed the MLPD+D ML features by up to 3.75% correct and 6.01% accuracy.

As an additional experiment, the scores from the GMM- and MLP-based AF detectors were fused on the TIMIT database. Two different methods of performing score fusion using an MLP were considered. The best detection performance was obtained using the Fusion 2 method. Feature sets were created using the outputs of the fusion MLPs and evaluated on a phone recognition task. Compared to an MFCC-based system, the Fusion 2 feature set yielded an increase of 4.26% correct and 3.78% accuracy when using context-independent models; with context-dependent models, the Fusion 2 feature set yielded an increase in percent correct of 2.84% and a difference in percent accuracy of 0.28%.

6.1 Future Work

Future work will first investigate using different feature vectors as input to the MLPwC-based AF detectors. For example, in Section 3.1 several features were mentioned that have been found useful for detecting voicing status and place of articulation. Formant estimation may also be useful as a feature for detecting some AFs. Furthermore, there may be a potential for increasing detection performance by simply changing the window size used when pre-processing the speech signal before calculating the MFCCs.

Incorporating context was found to increase the accuracy and reduce the EER of the AF detectors created using MLPs; thus including context in the GMM-based AF detectors may also yield

an increase in detection performance. This could be accomplished by training the GMMs using a single feature vector formed by concatenating the feature vectors calculated from a series of speech frames.

Another possibility for improving recognition performance is combining the proposed feature sets with MFCCs. Combining the scores from the GMM-based AF detectors with MFCC context-dependent HMMs at the log-likelihood level when scoring has been shown to yield a decrease in WER [25]. The feature vectors could also be concatenated and processed with a KLT or LDA for use as a single feature set.

Bibliography

[1] J. Clark and C. Yallop, *An Introduction to Phonetics and Phonology*, 2nd ed. Massachusetts: Blackwell, 1995.

[2] International Phonetic Association, *Handbook of the International Phonetic Association*. Cambridge University Press, 1999.

[3] S. Stevens and J. Volkmann, "The Relation of Pitch to Frequency: A Revised Scale," *American Journal of Psychology*, vol. 153, pp. 329–353, 1940.

[4] B. Atal, "Effectiveness of Linear Prediction Characteristics of the Speech Wave for Automatic Speaker Identification and Verification," *The Journal of the Acoustical Society of America*, vol. 55, pp. 1304–1312, 1974.

[5] J. Wilpon, C. Lee, and L. Rabiner, "Improvements in Connected Digit Recognition Using Higher Order Spectral and Energy Features," in *Proceedings of ICASSP*, vol. 1, pp. 349–352, 1991.

[6] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," in *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.

[7] L. Rabiner and B.-H. Juang, *Fundementals of Speech Recognition*. New Jersey: Prentice-Hall PTR, 1993.

[8] X. Haung, A. Acero, and H.-W. Hon *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. New Jersey: Prentice-Hall PTR, 2001.

[9] Cambridge University Engineering Department, "The HTK Book," 2007. (Available at http://htk.eng.cam.ac.uk).

[10] L. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Mathematical Statistics*, vol. 41, pp. 164–171, 1970.

[11] J. Blimes, "A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models", International Computer Science Institute, University of California at Berkely, TR-97-021, 1998.

[12] S. Young, J. Odell, and P. Woodland, "Tree-Based State Tying for High Accuracy Acoustic Modeling," in *Proceedings of the ARPA Workshop on Human Language*, pp. 307–312, 1994.

[13] H. Ney, U. Essen, and R. Kneser, "On Structuring Probabilistic Dependences in Stochastic Language Modeling," *Computer Speech and Language*, vol. 8, pp. 1–38, 1994.

[14] I. Good, "The Population of Species and the Estimation of Population Parameters," *Biometrika*, vol. 40, pp. 237–264, 1953.

[15] I. Witten and T. Bell, "The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression," *IEEE Transcriptions on Information Theory*, vol. 37, pp. 1085–1094, 1991.

[16] S. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 35, pp. 400–401, 1987.

[17] S. Young, "Large Vocabulary Continuous Speech Recognition: A Review," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 3–28, 1995.

[18] Y. Akgul, C. Kambhamettu, and M. Stone, "Automatic Extraction and Tracking of the Tongue Contours," *IEEE Transactions on Medical Imaging*, vol. 18, pp. 1035–1045, 1999.

[19] B. Branderund, H.-J. Lundberg, H. Djamshidpey, I. Wäneland, D. Krull, and B. Lindblom, "X-Ray Analyses of Speech: Methodological Aspects," in *Proceedings of FONETIK-1998*, Stockholm University, pp. 168–171, 1998.

[20] O. Engwall, "Dynamic Aspects of Coarticulation in Swedish Fricatives-a Combined EMA and EPG Study", TMH-QPSR4, pp. 49–73, 2000.

[21] A. Abdelatty Ali, J. Spiegel, and P. Mueller, "Robust Classification of Stop Consonants using Auditory-Based Speech Processing," presented at ICASSP-2001, Salt Lake City, USA, 2001.

[22] D. Kocharov, A. Zolnay, R. Schüter, and H. Ney, "Articulatory Motivated Acoustic Features for Speech Recognition," in *Interspeech-2005 Proceedings*, Libson, Portugal, pp. 1101–1104, 2005.

[23] K. Kirchhoff, "Robust Speech Recognition Using Articulatory Information," PhD thesis, University of Bielefeld, Germany, 1999.

[24] O. Scharenborg, V. Wan, and R. Moore, "Capturing Fine-Phonetic Variation Through Automatic Classification of Articulatory Features," in *Speech Recognition and Intrinsic Variation (SRIV-2006)*, Toulouse, France, pp. 77–82, 2006.

[25] S. Stüker, "Multilingual Articulatory Features," Diplomarbeit, Universität Fridericiana zu Karlsruhe, Germany, 2003.

[26] J. Morris and E. Fossler-Lussier, "Combining Phonetic Attributes Using Conditional Random Fields," in *Proceedings of ICSLP-2006*, Pittsburgh, USA, 2006.

[27] J. Frankel, M. Wester, and S. King, "Articulatory Feature Recognition Using Dynamic Bayesian Networks," in *Proceedings of ICSLP-2004*, Jeju Island, South Korea, 2004.

[28] Linguistic Data Consortium, *CSR-II (WSJ1) Complete*, 1994. (Available at http://www.ldc.upenn.edu).

[29] T. Schultz, "GlobalPhone: A Multilingual Speech and Text Database Developed at Karlsruhe University," in *Proceedings of ICSLP-2002*, pp. 345–348, 2002.

[30] D. Reynolds, T. Quartieri, and R. Dunn, "Speaker Verification using Adapted Gaussian Mixture Models," *Digital Signal Processing*, vol. 10, pp. 19–41, 2000.

[31] C. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford University Press, 1995.

[32] International Computer Science Institute, *ICSI QuickNet Software Package*, 2006. (Available at http://www.icsi.berkeley.edu/Speech/qn.html).

[33] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, The DET Curve in Assessment of Detection Preformance," in *Proceedings of Eurospeech-1997*, pp. 1895–1898, 1997.

[34] National Institute of Standards and Technology, *DETware*, 2007. (Available at http://www.nist.gov/speech/tools/index.htm).

[35] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, and N. Dahlgran, "DARPA TIMIT Acoustic Phonetic Continuous Speech Corpus CDROM," 1993.

Appendix A

AF Detection Accuracy

AF	English	German	Spanish	Japanese
CONSONANT	88.00%	86.95%	84.72%	87.64%
bilabial	94.66%	92.64%	93.35%	96.35%
labiodental	97.05%	94.79%	98.59%	97.85%
labialvelar	98.05%	-	-	-
dental	97.41%	-	95.90%	-
alveolar	85.24%	80.35%	86.17%	87.85%
post alveolar	97.39%	95.66%	99.49%	95.44%
retroflex	95.85%	-	-	-
palatal	98.47%	97.92%	95.90%	97.81%
velar	95.49%	93.28%	91.66%	93.29%
uvular	-	-	-	96.27%
glottal	98.69%	98.24%	-	95.61%
plosive	91.42%	89.78%	90.45%	90.47%
nasal	95.69%	94.89%	94.94%	94.73%
trill	-	93.58%	98.63%	-
tap or flap	98.98%	-	96.15%	-
fricative	92.80%	91.19%	92.61%	94.34%
approximant	93.83%	99.02%	97.22%	96.50%
lateral approximant	96.70%	96.99%	94.95%	97.37%
affricate	96.15%	96.49%	99.49%	95.31%
voiced	90.93%	86.42%	92.61%	93.09%
unvoiced	89.95%	83.03%	89.46%	91.49%
VOWEL	89.95%	89.54%	89.17%	89.59%
close	95.29%	90.54%	93.25%	91.98%
near-close	96.23%	-	-	-
close-mid	94.68%	90.96%	89.63%	92.13%
mid	91.79%	92.68%	-	-
open-mid	91.82%	98.57%	-	-
near-open	96.79%	96.65%	-	-
open	97.37%	95.64%	95.35%	95.96%
front	93.86%	90.05%	92.66%	93.83%
near-front	96.67%	-	-	-
central	91.02%	91.39%	-	-
near-back	99.21%	-	-	-
back	93.80%	95.54%	92.77%	91.75%
rounded	94.74%	91.01%	95.05%	95.16%
unrounded	90.75%	90.67%	90.88%	90.11%
stress	-	-	93.84%	-
silence	97.58%	96.99%	94.42%	96.00%

Table A.1: Monolingual GMM-based AF detection accuracy

AF	English	German	Spanish	Japanese
CONSONANT	89.19%	86.62%	85.36%	88.84%
bilabial	95.92%	94.16%	94.49%	97.51%
labiodental	97.86%	95.88%	99.27%	98.84%
labialvelar	99.16%	-	-	-
dental	98.61%	-	97.70%	-
alveolar	85.97%	81.13%	87.95%	89.24%
post alveolar	98.36%	96.59%	99.74%	96.75%
retroflex	97.60%	-	-	-
palatal	99.11%	98.53%	97.35%	98.64%
velar	96.32%	95.06%	93.89%	94.63%
uvular	-	-	-	97.36%
glottal	99.35%	99.33%	-	97.27%
plosive	92.98%	91.15%	91.19%	91.73%
nasal	95.82%	95.05%	94.18%	95.73%
trill	-	96.04%	99.39%	-
tap or flap	99.45%	-	97.40%	-
fricative	93.51%	91.48%	93.69%	95.29%
approximant	95.76%	99.66%	98.50%	97.79%
lateral approximant	97.60%	97.34%	95.66%	98.44%
affricate	98.15%	97.59%	99.74%	97.13%
voiced	91.91%	87.13%	92.99%	93.74%
unvoiced	91.10%	87.48%	89.49%	92.70%
VOWEL	90.41%	89.69%	89.73%	90.46%
close	96.65%	92.23%	95.33%	94.10%
near-close	98.18%	-	-	-
close-mid	96.40%	92.96%	90.85%	93.85%
mid	94.77%	95.89%	-	-
open-mid	95.03%	99.61%	-	-
near-open	98.18%	98.80%	-	-
open	98.59%	95.08%	95.20%	96.53%
front	94.46%	89.26%	93.99%	94.73%
near-front	98.36%	-	-	-
central	93.99%	94.63%	-	-
near-back	99.82%	-	-	-
back	95.24%	96.31%	93.15%	92.59%
rounded	96.11%	93.23%	94.88%	96.63%
unrounded	91.42%	89.57%	91.05%	91.12%
stress	-	-	97.74%	-
silence	97.91%	96.89%	94.79%	96.52%

Table A.2: Monolingual MLP-based AF detection accuracy

AF	English	German	Spanish	Japanese
CONSONANT	89.68%	87.61%	88.53%	90.47%
bilabial	96.83%	95.30%	95.61%	98.10%
labiodental	98.08%	96.71%	99.46%	98.90%
labialvelar	99.36%	-	-	-
dental	98.73%	-	97.82%	-
alveolar	86.51%	83.03%	90.23%	92.25%
post alveolar	98.56%	97.34%	99.82%	97.19%
retroflex	97.88%	-	-	-
palatal	99.17%	98.85%	97.83%	98.85%
velar	97.02%	95.68%	94.68%	95.88%
uvular	-	-	-	98.15%
glottal	99.49%	99.41%	-	97.42%
plosive	94.31%	93.14%	93.70%	93.38%
nasal	96.25%	95.42%	95.42%	96.39%
trill	-	96.37%	99.54%	-
tap or flap	99.53%	-	97.94%	-
fricative	94.49%	93.21%	93.89%	96.21%
approximant	96.20%	99.69%	98.79%	97.96%
lateral approximant	97.91%	97.67%	96.60%	98.77%
affricate	98.36%	98.26%	99.82%	97.96%
voiced	92.88%	89.30%	93.05%	94.69%
unvoiced	92.77%	90.39%	92.59%	93.97%
VOWEL	91.34%	90.52%	90.95%	91.63%
close	96.83%	92.66%	95.72%	94.98%
near-close	98.35%	-	-	-
close-mid	96.51%	93.01%	91.73%	94.62%
mid	95.09%	96.21%	-	-
open-mid	95.30%	99.62%	-	-
near-open	98.33%	98.79%	-	-
open	98.67%	95.36%	95.67%	96.98%
front	95.11%	90.69%	94.07%	95.27%
near-front	98.53%	-	-	-
central	94.51%	95.08%	-	-
near-back	99.83%	-	-	-
back	95.26%	96.33%	93.46%	93.66%
rounded	95.74%	93.25%	95.28%	97.10%
unrounded	92.11%	90.52%	92.15%	92.21%
stress	-	-	97.59%	-
silence	98.51%	98.32%	96.85%	97.91%

Table A.3: Monolingual MLPwC-based AF detection accuracy

AF	English	German	Spanish	Japanese
CONSONANT	88.25%	87.55%	86.32%	87.19%
bilabial	96.67%	95.56%	95.72%	97.53%
labiodental	98.23%	97.09%	99.28%	98.26%
labialvelar	-	-	-	-
dental	98.59%	-	97.84%	-
alveolar	84.98%	81.37%	87.94%	87.92%
post alveolar	98.33%	96.49%	99.65%	95.96%
retroflex	-	-	-	-
palatal	99.17%	98.19%	97.50%	98.29%
velar	96.72%	95.15%	94.51%	95.25%
uvular	-	-	-	-
glottal	99.45%	99.30%	-	97.87%
plosive	93.39%	92.72%	93.16%	92.92%
nasal	96.77%	95.79%	96.06%	96.01%
trill	-	96.38%	99.34%	-
tap or flap	99.41%	-	98.06%	-
fricative	94.02%	92.80%	93.92%	95.11%
approximant	95.79%	99.56%	98.50%	97.60%
lateral approximant	97.05%	97.80%	96.38%	98.06%
affricate	98.31%	98.04%	99.68%	97.77%
voiced	91.48%	87.18%	92.57%	92.88%
unvoiced	91.54%	89.15%	91.33%	92.56%
VOWEL	89.72%	89.29%	89.48%	88.71%
close	95.87%	92.39%	94.96%	92.68%
near-close	-	-	-	-
close-mid	94.86%	92.34%	90.67%	93.24%
mid	95.04%	96.21%	-	-
open-mid	94.90%	99.27%	-	-
near-open	98.35%	98.67%	-	-
open	97.28%	96.36%	95.86%	96.16%
front	93.22%	90.05%	92.89%	91.65%
near-front	-	-	-	-
central	94.39%	95.01%	-	-
near-back	-	-	-	-
back	92.81%	93.67%	92.34%	89.82%
rounded	95.47%	92.78%	95.41%	96.00%
unrounded	90.17%	90.48%	90.14%	88.40%
stress	-	-	-	-
silence	98.35%	98.24%	96.39%	97.65%

Table A.4: Multilingual MLPwC-based AF detection accuracy