UNIVERSITY OF CAPE TOWN

Feature Extraction and Normalization in SVM Speaker Verification Using Telephone Speech

Thembisile Thulisile Mazibuko
Department of Electrical Engineering
University of Cape Town
South Africa
June 2007

Thesis presented for the degree of Master of Science in Engineering to the Department of Electrical Engineering, University of Cape Town.

DECLARATION

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own.

This work is being submitted for the degree of Master of Science in Engineering to the Electrical Engineering Department at the University of Cape Town. It has not been submitted for any degree or examination at any other university.

Author's Signature:.. Signed by candidate

Date:.....June 2007.....

ABSTRACT

In this research the Support Vector Machine classifier is applied to a text independent speaker verification task using conversational telephone speech from the NIST 2000 Speaker Recognition Evaluation. The SVM is a discriminative classifier with good generalization characteristics. It has been shown to perform as well as, and sometimes outperform the more widely used Gaussian Mixture Model. The SVM, like other classifiers is vulnerable to environmental noise, distortions from transmission over communication channels such as the telephone channel, and intersession variability.

Feature normalization is applied to compensate for the distortions that occur to the signal due to transmission over the telephone channel and mismatch between training and testing conditions. The performance achieved by applying Histogram Equalization is compared to that of common linear feature normalization techniques Cepstral Mean Subtraction and Mean Variance Normalization. The results show an improvement in SVM classification performance when feature compensation is applied, with HEQ achieving the best performance.

Although it has good generalization and classification performance, the SVM suffers the distinct disadvantage of very long computation time, especially in the training phase. Part of this thesis is thus dedicated to the application of feature extraction and dimensionality reduction techniques to reduce the training time of the SVM. The application two linear independent techniques, namely of Independent Component Analysis and Principal Component Analysis, is considered.

ICA often has difficulty reaching convergence and is unable to match the reduction to training time achieved by PCA. With the data dimensionality reduced to 30% of the original, PCA feature extraction is able to match the baseline performance while reducing the average training time by almost 80%. HEQ feature normalization however results in a greater reduction of the average SVM training time as well as better accuracy. When HEQ is applied a 90% reduction in average SVM training time is achieved as well as a

47% decrease in testing time. Applying HEQ also results in n improvement to the baseline EER where there is a mismatch in training and testing handset times.

The results show that SVM classification is sensitive to the data representation and the pre-processing that is applied to the feature prior to training or testing the SVM affects not only the accuracy of the classifier but also the computation time as well as the storage resources required by the SVM models.

Key words: speaker verification, support vector machine, feature extraction, feature normalization

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ANN – Artificial Neural Network

CMS – Cepstral Mean Subtraction

CMN – Cepstral Mean Normalization

DET – Detection Error Trade-off

DCF – Detection Cost Function

EER – Equal Error Rate

ERM – Empirical Risk Minimization

FAR – False Acceptance Rate

FRR – False Rejection Rate

GMM – Gaussian Mixture Model

HEQ – Histogram Equalization

HLT – Human Language Technology

HMM – Hidden Markov Model

HN – Histogram Normalization

ICA – Independent Component Analysis

LLR – Log Likelihood Ratio

LP – Linear Prediction

LPC – Linear Predictive Coding

LPCC – Linear Predictive Coding Cepstral

MFCC – Mel-frequency Cepstral Coefficients

MVN – Mean Variance Normalization

PCA – Principal Component Analysis

SID – Speaker Identification

SLT – Statistical Learning Theory

SRM – Structural Risk Minimization

SVM – Support Vector Machine

SV – Speaker Verification

VAD – Voice Activity Detection

CHAPTER ONE

1. INTRODUCTION

Speech is arguably the most natural mode of communication amongst the majority of the world's population. The fact that no special skill (e.g. the ability to write or type) or specialised equipment (e.g. a typewriter or personal computer) is required to participate in speech communication makes it easy and non-threatening. The development of the telephone network has meant that speech based communication is not restricted to communication with people who are within earshot. This has lead to a growth in the development of technological solutions which encompass human speech.

A large part of Human Language Technology (HLT) is concerned with the creation and development of Spoken Language Interfaces (SLI) to computers that allow humans to communicate with machines using speech. The technologies embodied in these interfaces include Speech Recognition, Speech Synthesis, Speaker Recognition, and Language Recognition [1]. The main focus of this thesis is Speaker Verification which is a component of Speaker Recognition. Figure 1.1 provides a graphical representation of the technologies which are included in HLT and gives a simplistic representation of how speaker verification relates to other technologies in this grouping.

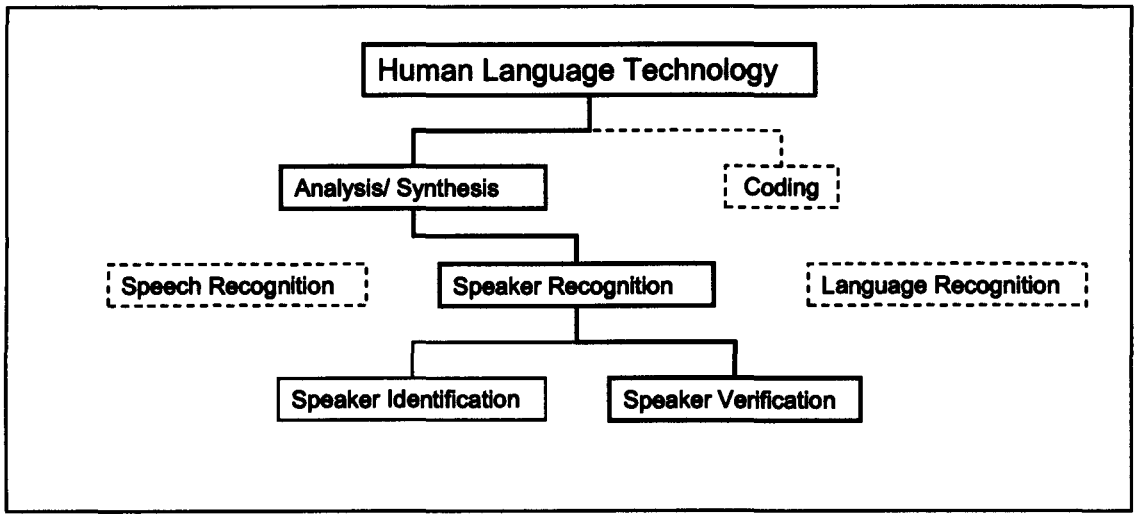

Figure 1.1: Human Language Technology

Speaker Recognition refers to the use of a machine to recognize a person from a spoken phrase [2]. There are two categories of speaker recognition, namely *speaker identification* (SID) and *speaker verification* (SV). The speaker identification task requires that, given an utterance, a decision is made regarding who is talking from a set of known speakers. *Open-set* SID considers the possibility of the speaker not belonging to the known set of speakers while *closed-set* SID does not consider that possibility. In Speaker Verification the task is, given an utterance as well as an identity claim, to decide whether or not the claimed identity matches the speech input, thus verifying the claim. That is, whether or not the speaker is who they claim to be. Both SID and SV can be text-dependent or text independent.

In text-dependent tasks the speaker is required to say a prescribed piece of text. Text-dependent systems generally give very good results but have the disadvantage of relying on the speaker's co-operation and require the user to remember the required phrase [2]. There is also the risk of an impostor using a recording of the authorised speaker to fool the system. Although this can be circumvented by random-phrase prompting, the flexibility of the system is still limited.

The text independent task is more flexible than the text dependent task since the user is not required to remember any predetermined text, but it is also the more challenging of the two tasks.

The focus of this thesis will be text-independent speaker verification. The next section discusses some of the issues concerning the reliability of speaker recognition systems. A discussion of a generic design of a speaker verification system is included and is followed by a review of some of the factors which affect the performance of speaker verification. The research objectives and the contributions to knowledge made by this research are then presented.

1.1. The Reliability of Speaker Recognition

A general misconception that exists is that speech is an unreliable biometric. This is largely because, to the human ear, people sometimes 'sound the same' and voices are easily imitated. However, as pointed out by Kinnunen in [3], this perception is based on subjective views of what other people sound like. In fact, speaker verification systems have been shown to perform competitively against human listeners in certain conditions [4, 5].

Voice is a combination of physiological and behavioural characteristics [6]. The physiological characteristics of human speech are determined by the shape and size of the vocal tract, mouth, nasal cavities and lips. These characteristics are largely invariant for each individual. The behavioural or learnt aspects of speech can, however, change frequently due to illness, emotional state, age etc. Voice is thus a different biometric to other biometrics such as the fingerprint or the retina in the sense that while the latter remain the same, the speech signal varies from time to time [3].

Reynolds [7] identifies two main factors that make speech a compelling biometric:

- Speech is a natural signal to produce and is not considered intrusive or inconvenient for users to provide.

- The use of the telephone network makes it easy to obtain and deliver speech signal for different applications.

The collection of speech is easy from the system viewpoint, and not threatening to the user. The comparison of speaker recognition to other biometric systems can be misleading simply because other biometric features have been the topics of research for much longer than the speech features. Fingerprint features have, for instance, been studied for almost 300 years longer than speech features [3]. Thus in comparing the results achieved by other biometrics, which might seem more impressive, to the performance of speech as a biometric, it is important to take into the account that the development of speech-based biometric systems is a young and on-going field of research.

1.2. Design of a Generic Speaker Verification System

Speaker verification systems generally operate in two modes; training/learning and testing/recognition [2]. Typically, the training mode comprises of: *data acquisition, feature analysis* and *speaker modelling* while the testing mode replaces the speaker modelling component with a *decision-making/classification* component. A graphical representation of this is shown in Figure 1.2.

Basically, the data acquired in the testing mode is compared to the speaker model generated during training and, based on the scoring function of the decision-making module; the user is either accepted by the system as matching the claimed identity or rejected as an impostor.

The data-acquisition and feature analysis modules are together referred to as the *front-end* of the system while the speaker modelling and decision-making components are collectively the system *back-end.*

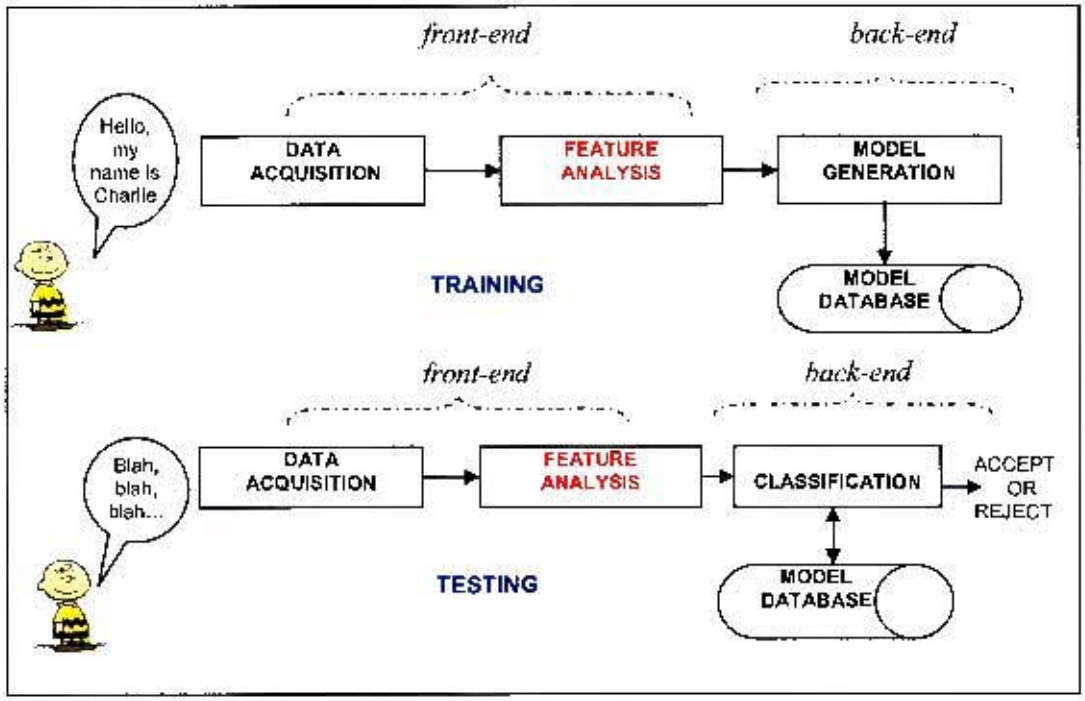

Figure1.2: A generic speaker verification system design

Input data to any recognizer or feature analysis module consist of a mix of relevant and irrelevant information [8]. The purpose of the feature analysis module is to remove the irrelevancies from the data and return a representation of the data which makes it easier to discriminate between different speakers. For instance, in the text-independent speaker verification paradigm, information regarding the language or content of the speech input might be deemed irrelevant while information regarding the gender of the speaker might be deemed vital to the task.

1.3. Problem Statement

The Support Vector Machine (SVM) [9] is a relatively new and powerful discriminative classification algorithm which has been successfully employed in HLT applications [10-13]. It has been shown to perform as well, and sometimes better than, the popular Gaussian Mixture Model (GMM) in speaker recognition and to improve the performance of generative models in hybrid systems [13, 14].

The work presented in this research focuses primarily on the application of the SVM classifier to the text-independent speaker verification task using telephone speech data.

The emphasis of this thesis is on two specific problems that arise in SVM-based speaker verification in the telephonic environment. The first of these is a general issue in speaker verification. Speaker recognition systems have generally been found to work very well in controlled, laboratory environments but suffer significant degradation in performance when applied in less-than-ideal conditions. Transmission of the speech signal over a telephone network is one such less-than-ideal scenario, as the channel distorts the signal, which often leads to degradation in system performance. Research has also shown that any mismatch between training and testing conditions has an adverse effect on the accuracy of speech processing systems [15]. This mismatch could be a consequence of factors such as the use of different telephone networks to transmit the training and testing data or the use of mismatched handsets or microphones to collect the data. Feature based compensation aims to reduce the negative effect of mismatched conditions by normalizing the feature vectors.

Cepstral Mean Normalization (CMN) [16], Mean Variance Normalization (MVN), RASTA Speech Processing [17] and, most recently, Histogram Equalization (HEQ) [18] are feature-based compensation techniques which have been successfully applied to improve the robustness of speaker verification systems based on generative speaker modelling techniques. These feature normalization techniques have different effects on the speech data and affect the system performance differently. The work presented in this thesis compares the effects of some of these compensation methods on the performance of an SVM classifier.

The second issue related to SVM speaker verification that is broached in this research is that of the speed of the SVM classifier. Although the SVM has good generalization characteristics and has been shown to have good performance in pattern recognition tasks, the long training time which is associated with the SVM classifier is a major drawback to its use and acts as a limiting factor to its application. This is particularly the

case for applications which might require real-time processing. Although the problem of SVM speed in the test phase has largely been solved [19] , the speed (or lack thereof) of the SVM in the training phase remains an issue for research.

In [20] Wang proposed a possible solution to the problem of the SVM speed which he referred to as the Reduced Dimensional SVM (RDSVM). The RDSVM consists of applying feature extraction techniques to reduce the dimension of the data before proceeding with an SVM-based speech recognition task. The idea behind RDSVM is that by reducing the data dimension, the amount of computation required for the SVM model generation will be reduced thereby reducing the total training time required. Feature extraction techniques attempt to remove irrelevancies in the data. The feature extraction methods applied in [20] were Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). In this thesis, this idea will be explored further by considering the application of PCA and ICA to feature extraction in a speaker verification task.

The SVM, feature compensation and feature transformation techniques will be discussed in detail in later chapters.

1.4. Research Objectives

The main objectives of this thesis are:

1. Evaluate the effect of contemporary feature-based compensation techniques on the performance of the SVM classifier in the speaker verification task.

2. Evaluate the application of linear feature extraction when applied in an SVM-based speaker verification system.

1.5. Research Contribution

The primary aim of this project is the exploration of the application of the SVM to a text-independent speaker verification task. SVMs have successfully been applied to pattern recognition tasks, including speaker verification. However, the algorithm suffers the drawback of extremely long training times. An important contribution of this work is the evaluation of the effect of linear independent feature extraction on SVM processing time and accuracy. The feature transformation is applied for dimensionality reduction, based on the hypothesis that reducing the data dimensionality will reduce the SVM training time by reducing the overall number of calculations that are required to be performed within the SVM

Another important contribution made in this work is the determination of the effect of HEQ on the accuracy of the SVM speaker verification system. HEQ, traditionally an image processing technique, has been successfully applied in speaker verification systems employing the GMM.

1.6. Scope and Limitations

This thesis presents an evaluation of the effect of featured-based compensation techniques on the performance of an SVM-based speaker verification system. The effect of linear independent feature extraction on the performance of SVM is also evaluated. A comparison of SVM-based speaker verification to other approaches, such as the state-of-the-art GMM-based system, is beyond the scope of this thesis.

1.7. Thesis Organization

The remainder of this thesis is organised as follows:

- Chapter Two presents an introduction to the fundamentals of speaker verification including the speech production process and the signal processing required in speaker verification.
- In Chapter Three a description of Support Vector Machine classification is given as well as a review of some of the relevant literature. The limitations of the SVM classifier are also discussed with reference to the effect that these limitations have on the application of the SVM classifier to speaker verification tasks.
- Feature normalization techniques are applied in speech processing tasks as a means of lessening the effect of contamination of the speech signal on the system performance. This contamination could be due to environmental effects or the results of transmission of the signal over the telephone networks. Chapter Four provides a review of some of the normalization techniques, namely Cepstral Mean Normalization, Mean Variance Normalization, RASTA Speech Processing and Histogram Normalization.
- The SVM discriminative classifier has been shown to possess characteristics which make it appealing to the speaker verification paradigm. It has been shown to perform as well as, and sometimes better than, the state-of-the-art generative techniques and is designed to have good generalization. However, the long training time associated with the SVM is a major limitation. Evaluating the effect of dimensionality reduction of the performance and speed of the classifier is a key objective of this research. Chapter Five presents a review of Principal Component Analysis and Independent Component Analysis which are the feature extraction techniques that are applied to this task.
- Chapter Six consists of an outline of the experimental framework used in this thesis as well as descriptions of some of the software tools which were used.
- The results of the experimental work and analysis thereof are presented in Chapter Seven.

- Chapter Eight provides a summary of the work done and achievements of this research as well as a discussion of avenues for future research relating to the concepts explored in this research.

1.8. Chapter Summary

This chapter provided an introduction to the speaker recognition field. The objectives and aims of this research were discussed as well as the scope and limitations of the work presented in the remainder of the document. The following chapter discusses the fundamental concepts of speaker verification in order to allow the reader some basic understanding of the processes involved in speaker verification.

CHAPTER TWO

2. FUNDAMENTALS OF SPEAKER VERIFICATION

This chapter provides a review of some of the key concepts involved in speaker verification. First, an introduction to human speech production is given followed by an introduction to the speech signal processing techniques used in speaker verification.

2.1. Human Speech Production

In order to understand speaker verification an understanding of the process and characteristics of human speech production which make it possible to differentiate between speakers is necessary. The two main sources of speaker specific characteristics in speech production are physiological and behavioural characteristics. These are roughly categorised into low-level acoustic information such as the sound of the voice, which is related to the physiology of the speech production apparatus, and high-level cues such as accent and word usage which are behavioural or learned characteristics of speech [21]. Speaker individuality is a complex phenomenon which builds up from both the anatomy of the speaker's vocal organs as well as learned traits [3] .

There are three commonly recognized physiological components involved in the production of human speech [3, 8]:

- the lungs and trachea
- larynx
- vocal tract

In the speech production process the lungs and trachea control the loudness of the resultant speech but make very little audible contribution to speech [8].

The larynx is a complex system of cartilages and muscles containing and controlling the vocal chords. In the production of speech, the function of the vocal chords is to provide the excitation source for speech [8]. The actions of the vocal folds produce different types of speech i.e. voiced, unvoiced.

Voiced speech is produced by the quasi-periodic opening and closing of the vocal folds which generates a glottal wave. The glottal wave passes through the vocal tract [22], which can be modelled as an acoustic tube, and amplifies the energy around the resonant frequencies while attenuating the energy around the anti-resonant frequencies. The resonances are referred to as *formants* [22]. The formant frequencies are person dependent and can thus be helpful in differentiating one speaker from another. In the production of unvoiced sounds the vocal folds do not vibrate which results in a speech signal that is non-periodic and random in nature [18].

The vocal tract generally refers to the speech production organs above the larynx [see Figure 2.1]. In speech production, the function of the vocal tract is the *colouring and articulation of the voice* [8].

The speech signal carries person dependent information due to the largely unique configuration of the vocal tract and vocal folds of each person. As the acoustic wave passes through the vocal tract, its spectrum is affected by the cavity resonances of the vocal tract. These resonances depend on the shapes of the various regions of the vocal

tract. The vocal tract shape can thus be estimated from the shape of the spectrum [2]. Typical text-independent speaker recognition systems use features derived from a frequency analysis of the voice to infer the shape of the vocal tract and identify the speaker [10], as both the length and the shape of the vocal tract are unique to each individual [8].

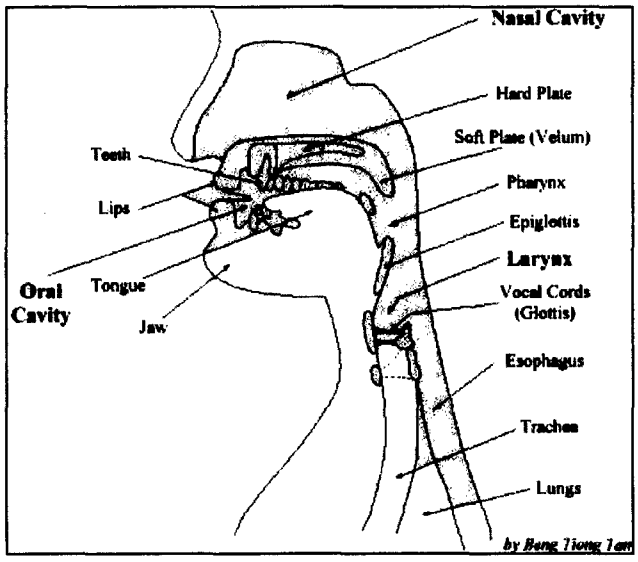

Figure 2.1: The human speech production system. [i]

The learnt or behavioural characteristics of speech include rate of speech, dialect and accent [10]. The bulk of speaker recognition research has focussed on modelling the physiological characteristics of speech. There has however been interesting work on the development of systems which employ the higher-level speech characteristics [21, 24-26].

Although high-level cues have been used successfully in speaker recognition tasks, the low-level spectral features have the advantage of easy computation and widespread use in speaker and speech recognition [3].

2.2. Speech Signal Pre-Processing

Rabiner and Juang [27] wrote about speech recognition: *...a good fundamental understanding of the way signal processing techniques are used to implement the parameter measurement phase of the recogniser is mandatory for understanding the strengths and shortcomings of the various approaches...that have been proposed and studied in literature.* The same can be said for speaker recognition. This section provides a review of the signal processing techniques employed in speaker verification.

The initial phase of the speaker recognition verification task is referred to as the front-end. Front-end analysis is the process whereby the input acoustic signal is converted into a sequence of acoustic feature vectors [28].

In order to be utilised in speaker verification systems, the analogue speech signal captured by the recording device (e.g. microphone) must be converted to a digitised format which can be used in computers and/or digital signal processors. This conversion is performed by an analogue-to-digital (A/D) converter. During this process the analogue signal $x(t)$ is sampled at some frequency f such that:

$$x[n] = x(nT) \tag{2.1}$$

where $T = \frac{1}{f}$ is the sampling period.

The Nyquist theorem requires that the sampling frequency be at least twice the bandwidth of the signal in question. That is, when sampling a signal with 4 kHz bandwidth, a sampling frequency of at least 8 kHz is required.

The precision with which the signal is quantised is determined by the number of bits used. The dynamic range of the signal is determined by the number of bits used per sample [3].

Speaker verification systems typically involve a signal pre-processing module. This module generally consists of the following subcomponents [2]:

- pre-emphasis filtering
- framing
- windowing
- voice activity detection

These elements are discussed briefly in the sections that follow.

2.2.1. Pre-Emphasis Filtering

The main function of the pre-emphasis filter is to *compensate for the spectral tilt that occurs as a result of voiced sounds having a steep roll-off in the high frequency region of the speech spectrum... this process results in a more balanced distribution of energy across the frequency range and greater distinction of the harmonics present in the signal* [2]. Pre-emphasis filtering thus aims to enhance the high frequencies of the spectrum, which are generally reduced by the speech production process [29].

It has been found that the numerical stability of linear predictive analysis is inversely proportional to the dynamic range of the signal being analyzed. Applying a filter such as a pre-emphasis filter, which flattens the spectrum prior to linear predictive analysis, therefore helps to avoid numerical problems where linear prediction is used [3].

A common transfer function for the pre-emphasis filter is [18]:

$$H(z) = 1 - \alpha z^{-1} \tag{2.2}$$

In the time domain the filter representation is given by [3]:

$$x_p(t) = x(t) - \alpha x(t-1) \tag{2.3}$$

The value of α is typically taken in the interval [0.95, 0.98].

2.2.2. Speech Framing

The speech signal changes continuously due to the movement of the vocal production organs [3]. However it tends to be fairly constant over short intervals and as a result it is common practise to process the signal in short segments referred to as frames [28]. This is referred to as *short-term spectral analysis*. The frames are selected to be in order of milliseconds. Typically the length of each frame is set to 20 or 30 milliseconds. The length of the frame is chosen to be long enough to allow adequate frequency resolution yet short enough to capture the local spectral properties [3]. The frames are overlapped as depicted in figure 2.2. The length of the overlap is typically 10 milliseconds. Each frame is represented by a feature vector which describes the average spectrum for a short time interval [28].

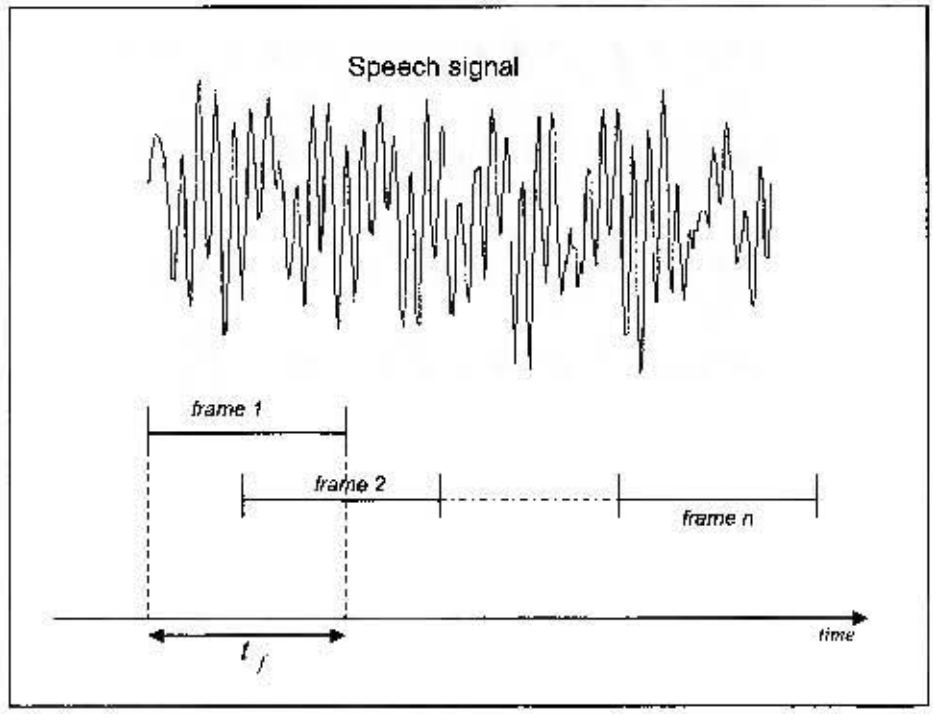

Figure 2.2: Framing the speech signal for short-term analysis.[ii]

2.2.3. Windowing

The purpose of windowing is to remove the effect of the spectral artefacts that result from the framing process and minimise the signal discontinuities at the start and end of each frame [3, 18]. These discontinuities introduce distortions to the signal frequency domain spectrum, primarily because of the implicit rectangular window employed in the framing process [18]. The most commonly used windowing function is the Hamming window [3, 27, 31, 32] and is defined as:

$$w[n] = \begin{cases} 0.54 - 0.46\cos((2n\pi)/N), & 0 \le n \le N-1 \\ 0 & , \quad \text{otherwise} \end{cases} \tag{2.4}$$

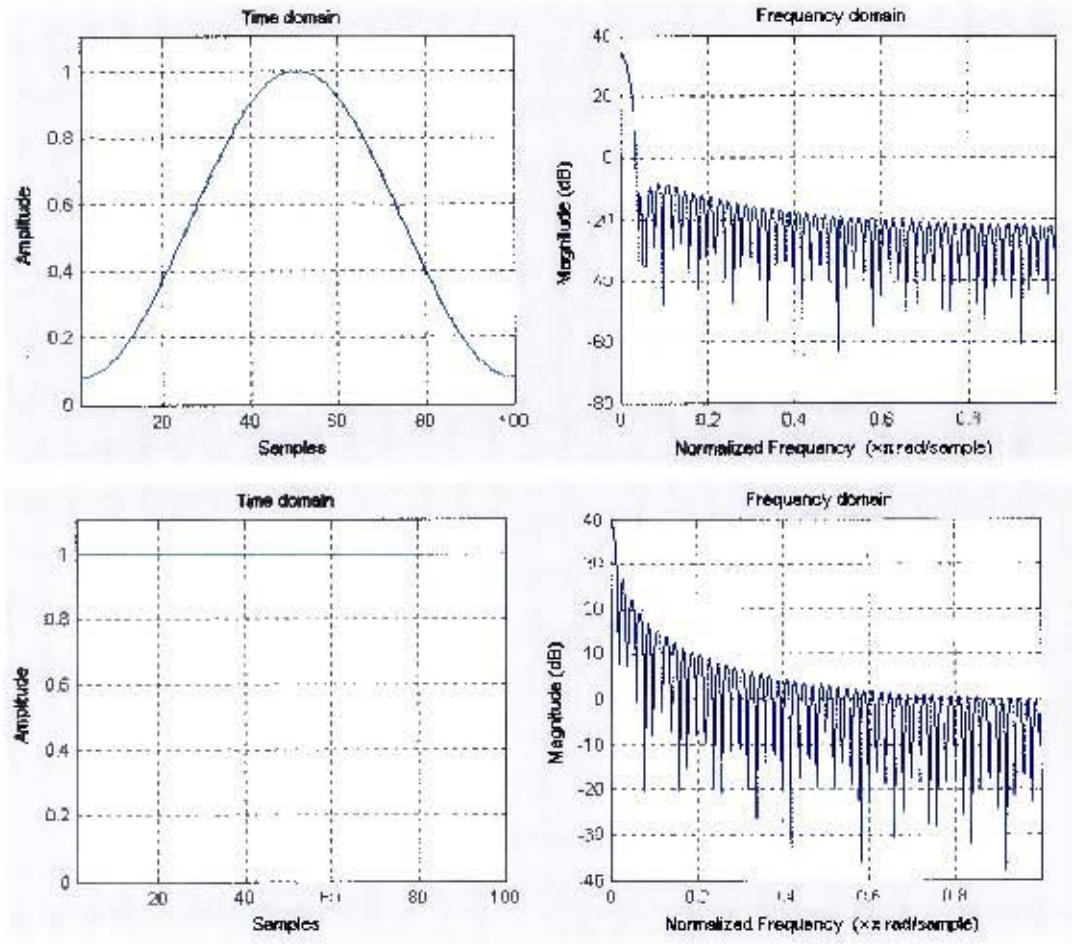

Figure 2.3: Time and frequency domain characteristics of the Hamming and rectangular windows

Figure 2.3 shows the characteristics of the rectangular and Hamming windows. The frequency domain representation of the rectangular window shows significantly more rippling in the stop-band [3].

2.2.4. Voice Activity Detection

Voice or Speech Activity Detection refers to the process of distinguishing the silent parts of an utterance from the portions that contain speech. The goal of the Voice Activity Detection (VAD) module is the removal of the irrelevant information that is carried in the silent parts of the signal.

There are generally two types of speech activity detection algorithms. The first category includes those algorithms that analyze the time variations of selected parameters (e.g. energy). These algorithms use a threshold to determine whether a portion of the speech signal is speech or non-speech [33]. The second category of VAD algorithms are based on pattern matching techniques. During training, reference models for two classes; speech and non-speech; are created based on selected speech features [33].

2.3. Speech Parameterization

The speech parameterization process aims at describing the significant information contained in the speech signal by an array of parameters referred to as speech features [34]. For speaker verification, features with high inter-speaker variability, low intra-speaker variability and high speaker discrimination powers are required [2]. Although there are no exclusive features conveying speaker identity in speech, source-filter theory of speech production says that the speech spectrum shape encodes information about the speaker's vocal tract shape via resonances or formants, and glottal source via pitch harmonics. Most speaker verification systems therefore use some form of spectral based features [7]. The two most common choices for the signal processing front-end are the Filterbank model and the Linear Predictive Coding (LPC) model [27].

The following sections discuss the speech parameterization methods based on LPC and Filterbank analysis.

2.3.1. Linear Predictive Coding

In linear prediction the present speech sample is predicted as a linear combination of past speech samples [35]. The LPC model analysis is performed on the speech frames with an all-pole modelling constraint [27]. Thus the resultant spectral representation $X_n(e^{jw})$ is constrained to the form $\sigma / A(e^{jw})$; where $A(e^{jw})$ is a p^{th} order polynomial whose z-transform is given by [27]:

$$A(z) = 1 + \alpha_1 z^{-1} + \alpha_2 z^{-2} + ... + \alpha_p z^{-p} \tag{2.5}$$

The basis of linear prediction is the assumption that adjacent samples of the speech waveform are highly correlated. Based on this, past samples can be used to make predictions about the signal behaviour [3]. The LP model assumes that each sample can be approximated by a linear combination of some past values [2, 3, 27]:

$$s(n) \approx \alpha_1 s(n-1) + \alpha_2 s(n-2) + ... + \alpha_k s(n-k) \tag{2.6}$$

$$s[n] = \sum_{k=1}^{p} \alpha[k] s[n-k] + Gu[n] \tag{2.7}$$

where p is the order of the predictor, $\{\alpha[k]; k = 1,..., p\}$ are the predictor coefficients and $Gu[n]$ is the excitation term ($u[n]$ is the excitation sequence and G its gain) [3]. The goal of LP analysis is to determine $\alpha[k]$ such that the average prediction error is minimized. This prediction error is also referred to as the *residual* and is given by [3]:

$$e[n] = s[n] - \sum_{k=1}^{p} \alpha_k s[n-k] \tag{2.8}$$

where $s[n]$ is the actual value of the nth sample. The residual is thus the difference between the actual and predicted sample values. As the residual becomes arbitrarily small, the predictor gives a very close approximation of the actual value [3].

The total squared prediction error is given by

$$E = \sum_n e[n]^2 \tag{2.9}$$

$$= \sum_n \left(s[n] - \sum_{k=1}^{p} \alpha_k s[n-k] \right) \tag{2.10}$$

which is minimized by setting the partial derivatives (with respect to the parameters $\alpha[k]$) to zero. The minimization is solved using either the *covariance method* or the *autocorrelation method* in order to find the optimal predictor coefficients [3, 18].

So far, only a time-domain representation of LP analysis has been considered. A frequency-domain representation of the problem is formulated next.

Considering (2.7) as the recurrence function equation of an infinite impulse response filter, with $\sum_{k=1}^{p} \alpha[k]s[n-k]$ representing the feedback part and $Gu[n]$ the input signal, $s[n]$ is then the result of the convolution of the scaled excitation signal with the filter [3]. The filter transfer function is

$$H(z) = \frac{S(z)}{U(z)} \tag{2.11}$$

$$= \frac{G}{1 - \sum_{k=1}^{p} \alpha[k] z^{-k}} \tag{2.12}$$

This is an all-pole filter. The poles of the transfer function model the envelope of the short-term spectrum and each pole represents a local peak in the magnitude spectrum.

This means that the model is restricted to only modelling the spectrum peaks, which are the resonances of the vocal tract [3].

The basic problem of LP analysis is the determination of a set of predictor coefficients, $\alpha[k]$ directly from the speech signal so that the spectral properties of the digital filter match those of the speech waveform within the analysis window [27].

A set of coefficients is estimated then the window is moved and a new estimation calculated [2]. The output of the LPC spectral analysis block is a vector of coefficients (LPC parameters) that parametrically specify the spectrum of an all-pole model that best matches the signal spectrum over the period of time over which the frame of speech samples was accumulated [27].

Linear predictive analysis has the advantage that it produces an estimate of the smoothed spectrum with most of the influence of the excitation removed [28]. The all-pole assumption imposed of the filter $H(z)$ means that the poles of $H(z)$ of are expected to be located at the formant frequencies, which suggests that the spectral envelope obtained through LP analysis is useful in formant estimation [3].

The LPC analysis method is mathematically precise and simple to implement. It provides a good model of the speech signal although more effective in the voiced regions of speech than the unvoiced and transient regions [3].

2.3.2. Filterbank Analysis

Filterbank analysis is a popular alternative to LPC. In this model the spectrum of the pre-processed speech frame is obtained by applying the Fourier transform [18, 22]. The spectrum is then filtered by passing it through a bank of bandpass filters whose coverage spans the frequency range of interest in the signal [27]. Each bandpass filter processes the speech signal independently to produce a spectral representation [27]. The aim of

Filterbank analysis is to obtain the average value of energy in a given frequency band [18, 27].

Typically in speaker recognition triangular filters are used and are arranged according to the mel-scale. The Mel-scale is an auditory scale that relates the perceived frequency to the actual frequency [29, 36]. Studies have found that the human ear is more sensitive to lower frequencies than it is to higher frequencies [37], and the application of the Mel-scale attempts to account for this in automatic speech processing. The filterbank is therefore arranged to compensate for this sensitivity to the lower frequencies. Figure 2.4 is a graphical representation of the arrangement of the filters in the filterbank according to the Mel-scale.

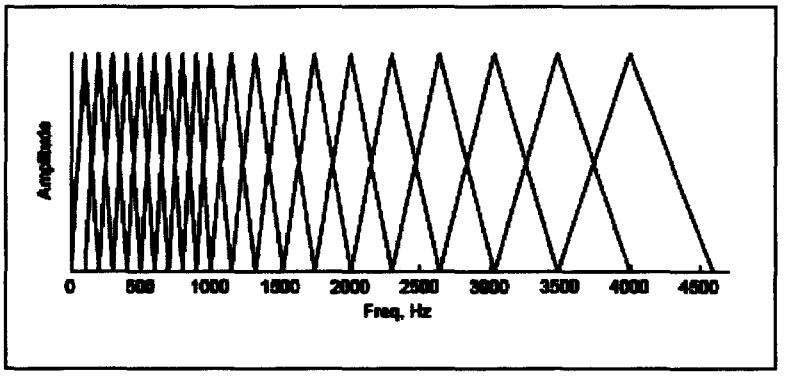

Figure 2.4: Mel-scaled filterbank.

A typical mathematical representation of the Mel-scale transformation is given by [18]:

$$f_{mel} = 2595 \cdot \log(1 + f_{linear}/700) \tag{2.13}$$

where f_{mel} is the Mel-scale frequency value and f_{linear} is the frequency value on the linear frequency scale. The value f_{mel} provides the location of the centre frequencies of each of the filters in the filterbank on the Mel-scale [18].

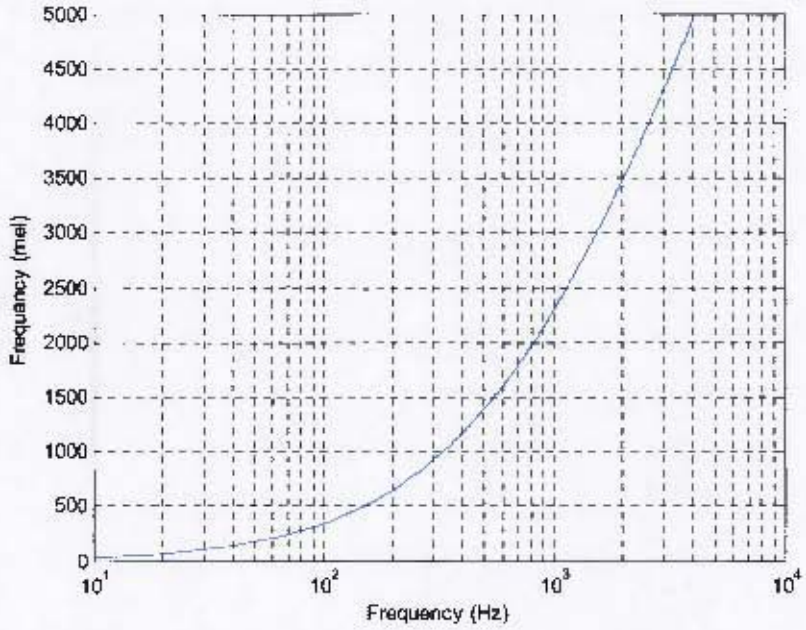

Figure 2.5: Conversion scale for Hertz to Mel-scale[1].

The Mel-scaled features have the advantage of taking human perception of speech into account.

2.4. Cepstral Analysis

In cepstral analysis the magnitude spectrum is represented as a combination of cosine basis functions with varying frequencies. The cepstral coefficients are the magnitudes of the basis functions [3].

The real cepstrum of a digital signal $s[n]$ is defined as *the inverse Fourier transform of the logarithm of the magnitude spectrum* [3]:

$$c[n] = \frac{1}{2}\pi \int_{\pi}^{\pi} \log\left|S\left(e^{j\omega}\right)\right| e^{j\omega n} d\omega \qquad (2.14)$$

[1] Hertz shown on a logarithmic scale

The coefficients $c[n]$ are the Fourier series coefficients of the log-spectrum where the lower cepstral coefficients represent the slow changes of the spectrum and the higher coefficients are the rapidly varying components of the spectrum [3].

Practically, the real cepstrum is obtained by applying the inverse discrete Fourier transform (IDFT) to the logarithm of the magnitude of the discrete Fourier transform (DFT) [3, 18].

2.4.1. Linear Predictive Coding Cepstral Coefficients

The recursion used for the derivation of the LPC cepstral coefficients is [27]:

$$c_o = \ln \sigma^2 \tag{2.15}$$

$$c_m = \alpha_m + \sum_{k=1}^{m-1} (k/m) c_k \alpha_{m-k} \quad 1 \le m \le p \tag{2.16}$$

$$c_m = \sum_{k=1}^{m-1} (k/m) c_k \alpha_{m-k} \quad m > p \tag{2.17}$$

where σ^2 is the gain term in the LPC model [27].

The cepstral coefficients are the coefficients of the Fourier transform representation of the log magnitude spectrum. These have been shown to be more robust than the LPC coefficients [27]. Although there are a finite number of LP coefficients, the LPC cepstrum sequence is infinite [3]. However, the magnitudes $|c_m|$ tend to zero quickly with increasing m, therefore a relatively small number of coefficients are needed to model the spectrum [3]. Generally, a cepstral representation with $q > p$ coefficients is used, with $q \cong \frac{3}{2} p$ [27].

The LPCC coefficients are derived from the predictor coefficients and as a result are subject to the all-pole assumption of the LPC model. The LPC cepstral coefficients are

therefore generally not the same as the cepstral coefficients derived directly from the magnitude spectrum [3].

2.4.2. Mel-Frequency Cepstral Features

The cepstral coefficients derived from filterbank analysis are referred to as *Mel-Frequency Cepstral Coefficients* (MFCC). In computing the Mel-cepstrum, the Mel-scaled filterbank is applied to the spectrum, then the inverse Fourier transform of the logarithm of the magnitude spectrum is computed [3]. The logarithm of the filterbank energies is then computed, and the discrete cosine transform (DCT) applied to these values to produce the cepstral coefficients [3, 18]:

$$c[n]=\sum_{k=1}^{K}\log(E_k)\cos\frac{\pi n}{K}\left(k-\frac{1}{2}\right) \qquad n=1,...,N \qquad (2.18)$$

where k is the number of filterbank outputs and N is the number of coefficients computed. The zero-th coefficient *c[0]* represents the log magnitude and thus depends on the intensity of the signal. It is therefore typically excluded from the feature set as a form of energy normalization [3, 18].

2.4.3. Dynamic Speech Features

In short-term spectral analysis each spectral vector is assumed to be short-term stationary. There is no time information encoded in the features obtained in the static features obtained from this analysis [3].

The articulatory movements that occur during speech are reflected in the measured spectrum as changes in formant frequencies and bandwidths [3]. The rate of these changes in the spectrum is dependant on the speaking style, speech rate and context. Some of these dynamic features are obviously characteristics of the speaker [3].

The optimum test to decide between the hypotheses is the Likelihood Ratio test which is given by [1]:

$$\frac{p(S|H_0)}{p(S|H_1)} \tag{2.19}$$

where $p(S|H_i)$ is the likelihood of hypothesis H_i given speech segment S. The decision threshold for accepting or rejecting a hypothesis is θ.

Generally a well defined model, estimated from the hypothesized speaker λ_{hyp}, is used to represent H_0. The likelihood ratio test also requires some model of the universe of non-claimant speakers which represents the alternative hypothesis H_1 [29, 38]. This model is represented by $\lambda_{\overline{hyp}}$ and is often referred to as the *background speaker model*.

There are two main approaches to generating background speaker models. The first approach uses a set of N other speakers, in the background speaker set, to approximate the alternative hypothesis such that the alternative hypothesis model is given by [29]:

$$p(X|\lambda_{\overline{hyp}}) = f(p(X|\lambda_1),..., p(X|\lambda_N)) \tag{2.20}$$

where $\{\lambda_1,...,\lambda_N\}$ is the set of N background speaker models.

Two issues that arise with the use of background speakers are [38]:

- the selection of speakers to use
- the number of speakers to use.

Typically this approach requires the use of speaker-specific background speaker sets in order to achieve the best performance [29]. This is an obvious limitation when there is a large number of users enrolled onto the system as each would require their own background speaker set.

The second approach is to train a single model (world model or universal background model) from speech collected from a large number of speakers [29]. The main advantage of this technique is that a single speaker-independent model is trained once and used for all the hypothesized speakers.

The models are estimated from the feature vectors created in the front-end processing described earlier in this chapter. Often the logarithm of the likelihood ratio is used giving the Log-Likelihood Ratio (LLR) [29]:

$$\Lambda(X) = \log p\left(X|\lambda_{hyp}\right) - \log p\left(X|\lambda_{\overline{hyp}}\right) \tag{2.21}$$

where X is the test feature vector .

2.6.1. Score Normalization

In the decision-making process of speaker verification the score obtained for a target model and test utterance pair is compared to some decision threshold and the test utterance is accepted as legitimate or rejected as an impostor based on the score. It has been found that there are often variances between the distribution of client/target scores and impostor scores during speaker verification tests [29]. This variability in test scores can make setting speaker-dependent decision thresholds a non-trivial task. Score normalization techniques aim to address the challenges created by the score variability.

The basis of score normalization is to apply the following normalization to each score generated [29]:

Let $L_\lambda(X)$ denote the score for speech signal X and speaker model λ. The normalized score is then given by

$$\tilde{L}_\lambda(X) = \frac{L_\lambda(X) - \mu_\lambda}{\sigma_\lambda} \tag{2.22}$$

where μ_λ and σ_λ are the normalization parameters for speaker λ which need to be estimated.

Some examples of score normalization techniques are zero normalization, handset normalization and test normalization. These techniques differ by the criteria they use to calculate the normalization parameters μ_λ and σ_λ.

2.7. Measuring System Performance

There are two types of errors that occur in the speaker verification process: *false rejections* and *false acceptances*. When the first type of error occurs the user is incorrectly refused access to the system. The other type of error occurs when an impostor is granted entry to the system. The rate at which these errors occur is often used to report on system performance.

$$\text{False Rejection Rate (FRR)} = \frac{\text{Number of clients rejected}}{\text{Total number of trials}} \tag{2.23}$$

$$\text{False Acceptance Rate (FRR)} = \frac{\text{Number of impostors accepted}}{\text{Total number of trials}} \tag{2.24}$$

The Equal Error Rate (EER), which occurs where FRR = FAR, is a common way of reporting system performance. A smaller EER value generally indicates a more accurate verification system.

The Detection Error Trade-off (DET) curve [43] is often used as a graphical representation of the performance of speaker verification systems. The DET curve is a plot of the two error type probabilities plotted on a *normal deviate* scale for varying

levels of the decision threshold. The closer to the origin the DET curve plot is, the better the system performance indicated.

The Detection Cost Function (DCF) is another measure of system performance. The DCF is a weighted sum of the false acceptance and false rejection error probabilities [19]:

$$DCF = C_{FalseReject} \times P_{Reject|Target} \times P_{Target} + C_{FalseAccept} \times P_{Accept|NonTarget} \times P_{NonTarget} \quad (2.25)$$

where $C_{FalseReject}$ refers to the cost associated with a false rejection error; $C_{FalseAccept}$ refers to the cost associated with a false acceptance error and P_{Target} is the *a priori* probability of the target. The choice of values for the parameters used in (2.25) is application dependent. When this performance measure is used the minimum DCF over all operating points is typically reported.

2.8. Chapter Summary

This chapter provided a review of the fundamental techniques and processes involved in speaker verification. Parameterization techniques Linear Predictive Coding and Filterbank analysis were discussed. Brief discussions of speaker modeling and score normalization techniques were also given.

The following chapter gives an introduction to the Support Vector Machine which is the speaker modeling technique that is used for this research.

CHAPTER THREE

3. SUPPORT VECTOR CLASSIFICATION

The Support Vector Machine (SVM) is a learning system that *uses a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimisation theory that implements a learning bias derived from statistical learning theory* [44]. The SVM is a supervised machine learning technique that learns the decision surface through a process of discrimination and is characterised by good generalization [12, 19].

The idea behind SVM classification can be expressed as follows [9, 45]: the input vectors $\mathbf{x}_i$ are mapped onto a high-dimensional space $\boldsymbol{H}$ through some nonlinear mapping,

$$\Phi : \mathbf{R}^N \rightarrow \mathbf{H} \tag{3.1}$$

The space $\boldsymbol{H}$ is commonly referred to as the feature space. The optimal separating hyperplane, which maximizes the margin, is then constructed in the high-dimensional space $\boldsymbol{H}$. This generates a nonlinear decision boundary in the input space. The main advantages of SVM classification are [46]:

- the SVM attempts to find a compromise between the minimization of the empirical risk and preventing over-fitting the data.
- the SVM problem is a convex quadratic programming problem and is readily solvable using quadratic programming techniques.

This chapter provides an introductory view at SVM formulation. First a discussion of Risk Minimization is given followed by a brief review of the formulation of linear and nonlinear decision boundaries for SVMs. The notation here largely follows that of [19] and [45].

3.1. Risk Minimization

Given a set of training data with l observations and each observation consisting of a pair $\{\mathbf{x}_i, y_i\}$ where $\mathbf{x}_i \in R^N, y_i \in \{-1,+1\}$ the task of the learning machine is to learn the mapping $\mathbf{x}_i \mapsto y_i$. That is, to find a function $f: R^N \to \{\pm 1\}$ such that f will correctly classify previously unseen examples.

The learning machine is defined by a set of possible mappings $\mathbf{x} \mapsto f(\mathbf{x},\alpha)$, where α represents adjustable parameters of the functions $f(\mathbf{x},\alpha)$. It is assumed to be deterministic, giving the same output $f(\mathbf{x},\alpha)$ for a given choice of α and input $\mathbf{x}$.

A particular choice of α generates a trained machine. The expected test error or risk for a trained machine is defined by:

$$R(\alpha) = \int \frac{1}{2}|y - f(\mathbf{x},\alpha)| \, dP(\mathbf{x},y) \qquad (3.2)$$

where $P(\mathbf{x},y)$ is the probability distribution generating the data and $\frac{1}{2}|y - f(\mathbf{x},\alpha)|$ is the loss. It is assumed that the test and training data are independently drawn by the same distribution.

Risk minimization involves minimizing equation (3.2). The distribution $P(\mathbf{x}, y)$ is often not known which makes this a non-trivial task. It is thus often simpler to minimize the *empirical risk* which is an approximation of the expected error and is defined by:

$$R_{emp}(\alpha) = \frac{1}{2l}\sum_{i=1}^{l}|y_i - f(\mathbf{x}_i, \alpha)| \tag{3.3}$$

Minimising equation (3.3) is referred to as Empirical Risk Minimisation (ERM). Note that the empirical risk is a fixed number for a particular choice of α and for a particular training set $\{\mathbf{x}_i, y_i\}$[19].

Due to the distribution of the training data being unknown, the choice of the number of parameters α is a challenge; too few parameters limit the performance of the classifier and too many parameters lead to over-fitting of the data and poor generalization of the classifier. Over-training is a result of the classifier having too many degrees of freedom which leads to learning irrelevant information in the data. The classifier is thus unable to generalize to unseen data. Good generalization performance of a classifier is achieved when the capacity of the classifying function and the training set size are matched [47]. A classifier with a large number of adjustable parameters, and therefore large capacity, is likely to learn a training set without error, but exhibit poor generalization, while a classifier with insufficient capacity might be completely unable to perform the learning task [47].

Cross-validation is a heuristic approach to avoiding over-fitting. An alternative approach is to quantify the complexity of the function $f(x, \alpha)$ and try to use functions with low complexity while still achieving a small empirical risk based on a theoretically derived upper bound for the true risk [48].

Structural Risk Minimization (SRM) consists of finding that subset of functions which minimizes the bound on the true risk [19]. Statistical Learning Theory (SLT) shows that

"...it is imperative to restrict the class of functions that f is chosen from to one which has a capacity that is suitable for the amount of training data...the minimization of these bounds, which depend on both the empirical risk and the capacity of the function class, leads to the principle of Structural Risk Minimization." [45]. The SRM method varies the classifier complexity in order to optimize the generalization ability [47].

The Vapnik-Chervonenkis (VC) dimension is the largest number of points which can be separated in all possible ways using functions of a given class [45]. If h is the VC dimension of a class of functions which can be implemented by the learning machine then, for all functions in that class, the bound [45]:

$$R(\alpha) \leq R_{emp}(\alpha) + \phi\left(\frac{h}{l}, \frac{\log(\eta)}{l}\right) \tag{3.4}$$

holds with probability $1 - \eta$ where ϕ is the confidence term or VC confidence given by [45]:

$$\phi\left(\frac{h}{l}, \frac{\log(\eta)}{l}\right) = \sqrt{\frac{h\left(\log\frac{2l}{h} + 1\right) - \log\left(\frac{\eta}{4}\right)}{l}} \tag{3.5}$$

It is important to note that this bound is completely independent of $P(\mathbf{x}, y)$. The assumption is that both the training and test data are drawn independently according to some $P(\mathbf{x}, y)$ [19].

The SVM formulation embodies the principles of SRM which has been found to be superior to ERM [49]. The SRM process minimises the upper bound on the expected error while ERM minimises the training error. This difference provides SVM with greater generalization ability [49].

3.2. Linearly Separable Case

The simplest case of SVM classification is that of linear machines trained on separable data [19].

Given training data $\{\mathbf{x}_i, y_i\}; \mathbf{x}_i \in R^N$ and $y_i \in \{-1,+1\}$ as before, suppose there is a hyperplane which separates the data classes. As Figure 3.1a shows, there is an infinite number of decision boundaries that could be chosen for separating the data. However, *among all hyperplanes separating data, there exists a unique one yielding the maximum margin of separation between the classes* [45]. This optimal separating hyperplane is *orthogonal to the shortest line connecting the convex hulls*[2] *of the two classes, and intersects this line halfway between the classes* [45].

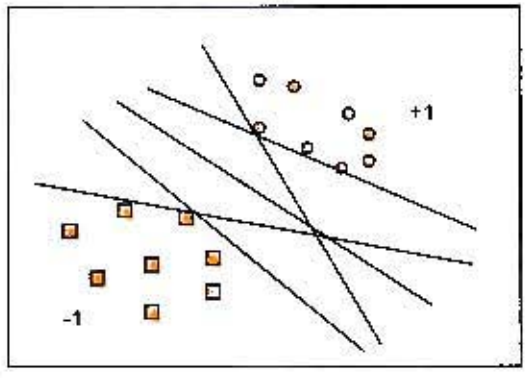

Figure 3.1a: There is an infinite number of separating planes.

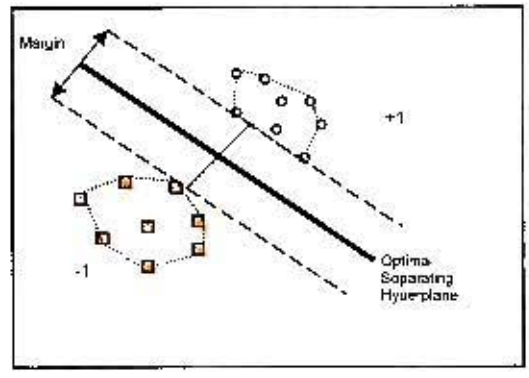

Figure 3.1b: The optimal separating hyperplane maximizes the margin.

Figure 3.1: Finding the optimal separating hyperplane.

Formally, we consider a class of hyperplanes $f(\mathbf{x},\mathbf{w}) = (\mathbf{w}\cdot\mathbf{x}) + b \quad \mathbf{w} \in R^N, b \in \mathbf{R}$. The points $\mathbf{x}$ which lie on the hyperplane satisfy $(\mathbf{w}\cdot\mathbf{x}) + b = 0$, where $\mathbf{w}$ is the normal to the hyperplane; $\frac{|b|}{\|\mathbf{w}\|}$ is the perpendicular distance from the hyperplane to the origin and $\|\mathbf{w}\|$ is the Euclidean norm of $\mathbf{w}$ [19]. For the linearly separable case the support vector

[2] A convex hull of a set of points is the smallest convex set that includes all the points.

algorithm looks for the separating plane which maximizes the margin. So, suppose all the data meet the constraints:

$$\mathbf{x}_i \cdot \mathbf{w} \geq +1 \text{ for } y_i = +1 \tag{3.6}$$

$$\mathbf{x}_i \cdot \mathbf{w} \leq -1 \text{ for } y_i = -1; \tag{3.7}$$

which combine to form the set of inequalities

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \forall i. \tag{3.8}$$

The perpendicular distance between the edges of the margin can be calculated by computation of the dot product between the unit vector perpendicular to the decision boundary and any vector from one edge of the margin to the other [10]. The points which lie on the two edges of the margin satisfy the equations:

$$\mathbf{x}_1 \cdot \mathbf{w} + b = +1 \tag{3.9}$$

$$\mathbf{x}_2 \cdot \mathbf{w} + b = -1. \tag{3.10}$$

The margin is therefore computed by

$$\frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot (\mathbf{x}_1 - \mathbf{x}_2) = \frac{2}{\|\mathbf{w}\|}. \tag{3.11}$$

Note that the capacity decreases with increasing margin [45].

The decision boundary which maximizes the margin also minimizes $\|\mathbf{w}\|^2$ subject to the inequality $(\mathbf{x}_i \cdot \mathbf{w} + b) y_i \geq 1$ for all i. This is an optimization problem which can be solved by introducing positive Lagrange multipliers $\alpha_i, i = 1..l$ for each of the inequality constraints in (3.8) and a Lagragian:

$$L(\mathbf{w},b,\alpha)=\frac{1}{2}\|\mathbf{w}\|^2-\sum_{i=1}^{l}\alpha_i(y_i\cdot((\mathbf{x}_i\cdot\mathbf{w})+b)-1) \tag{3.12}$$

$$=\frac{1}{2}\|\mathbf{w}\|^2-\sum_{i=1}^{l}\alpha_i y_i(\mathbf{x}_i\cdot\mathbf{w}+b)+\sum_{i=1}^{l}\alpha_i\,. \tag{3.13}$$

The Lagragian in (3.13) must be minimized in terms of the primal variables w and b, and maximized in terms of the dual variables α_i [45]. We require that the derivatives of L with respect to the primal variables vanish such that;

$$\frac{\partial}{\partial\mathbf{w}}L(\mathbf{w},b,\alpha)=0 \tag{3.14}$$

and

$$\frac{\partial}{\partial b}L(\mathbf{w},b,\alpha)=0\,. \tag{3.15}$$

This solves to

$$\mathbf{w}=\sum_{i=1}^{l}\alpha_i y_i\mathbf{x}_i \tag{3.16}$$

and

$$\sum_{i=1}^{l}\alpha_i y_i=0\,. \tag{3.17}$$

Substituting into (3.13), we eliminate the primal variables and arrive at the dual formulation of the optimization problem

$$L_D(\alpha)=\sum_i\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j\mathbf{x}_i\mathbf{x}_j\,. \tag{3.18}$$

Support vector training for the linearly separable case therefore amounts to maximising (3.18) subject to:

$$\alpha_i\geq 0,\quad i=1,\ldots,l \tag{3.19}$$

and

$$\sum_{i=1}^{l}\alpha_i y_i=0\,. \tag{3.20}$$

Note that there exists a Lagrange multiplier α_i for each point in the training data. In the solution, only those training points which lie on the hyperplanes defining the edges of the margin, and therefore satisfy (3.19) or (3.20) will have $\alpha_i > 0$. These points are referred to as the *support vectors*. All other training points will have $\alpha_i = 0$ and do not form part of the solutions. Figure 3.2 provides a simplified graphical representation of the linearly separable problem in SVM formulation.

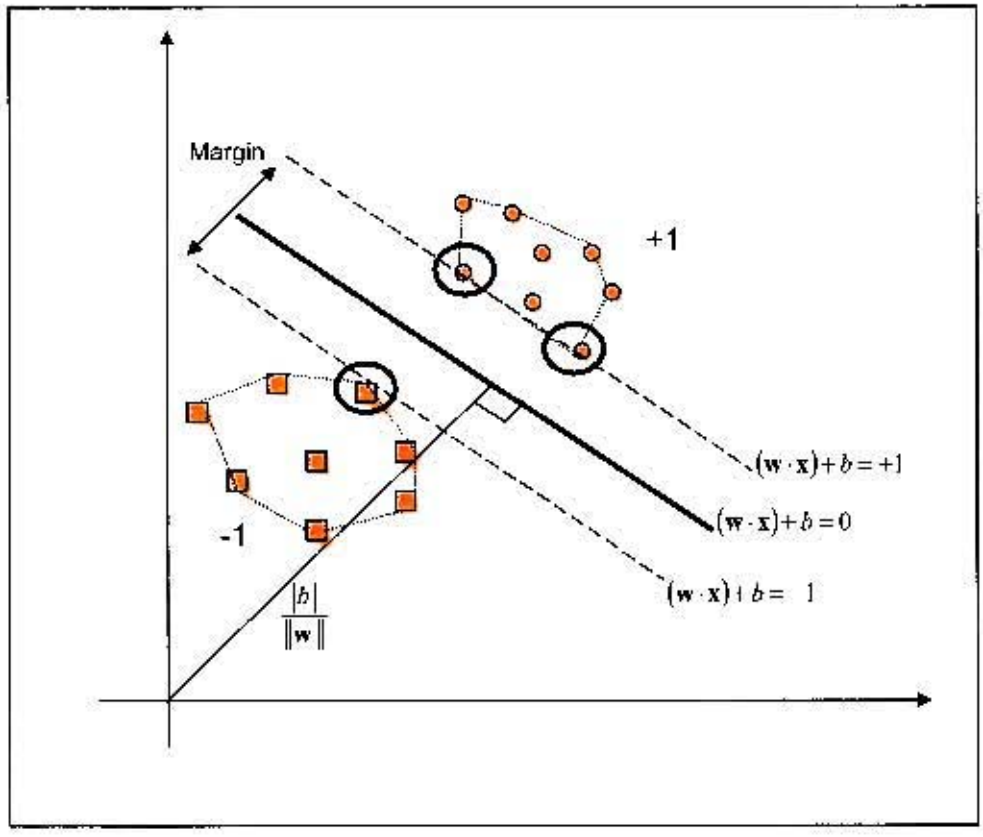

Figure 3.2: Optimal separating hyperplane for linearly separable data[3].

[3] The circled points are the support vectors.

The hyperplane decision function can be expressed as

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i \cdot (\mathbf{x} \cdot \mathbf{x}_i) + b\right). \tag{3.21}$$

The Karush-Kuhn-Tucker (KKT) conditions which, for the problem at hand, can be stated as:

$$\frac{\partial}{\partial w_k} L = w_k - \sum_i \alpha_i y_i x_{ik} = 0; \quad k = 1,\ldots,d \tag{3.22}$$

$$\frac{\partial}{\partial b} L = -\sum_i \alpha_i y_i = 0 \tag{3.23}$$

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0; \; i = 1,\ldots,l \tag{3.24}$$

$$\alpha_i \geq 0 \qquad \forall i \tag{3.25}$$

$$\alpha_i(y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1) = 0; \quad \forall i \tag{3.26}$$

are necessary and sufficient for w, b and α to be a solution [19]. Solving the support vector problem is therefore equivalent to finding a solution to the KKT conditions and using the KKT complementarity condition (3.25), we can solve for b using any i for which $\alpha_i \neq 0$ [19].

The support vectors are the critically important points in the training data. If training were repeated with all other training points removed or rearranged (but not so as to cross the margin-edge hyperplanes), the same separating hyperplane would be found [19].

3.3. Nonlinearly Separable Case

The extension of the above algorithm to the scenario where the data are not linearly separable is achieved by relaxing the constraints (3.6) and (3.7) and introducing a cost for doing so [19].

This is achieved by introducing positive slack variables $\xi_i, i = 1,...,l$ to the constraints so that these become

$$\mathbf{x}_i \cdot \mathbf{w} \geq +1 - \xi_i \text{ for } y_i = +1 \tag{3.27}$$

$$\mathbf{x}_i \cdot \mathbf{w} \leq -1 + \xi_i \text{ for } y_i = -1 \tag{3.28}$$

$$\xi_i \geq 0 \forall i. \tag{3.29}$$

An error occurs if the corresponding $\xi_i > 1$. The sum of all the slack variables $\sum_i \xi_i$ is then an upper bound on the number of training errors. The objective function to be minimized is modified to account for the extra cost of errors and changes to

$$\|\mathbf{w}\|^2 + C\left(\sum_i \xi_i\right) \tag{3.30}$$

where C is a user-specified parameter which corresponds to the penalty for errors. A larger value for C corresponds to assigning higher penalties for errors.

The dual optimization problem then requires the maximization of (3.17) subject to

$$0 \leq \alpha_i \leq C \tag{3.31}$$

$$\sum_i \alpha_i y_i = 0. \tag{3.32}$$

The solution is given by

$$\mathbf{w} = \sum_{i=1}^{ls} \alpha_i y_i \mathbf{x}_i \tag{3.33}$$

where *ls* is the number of support vectors. The significant difference from the linearly separable case is that C now acts as an upper bound to the α_i. A graphical representation of this scenario is shown in Figure 3.3.

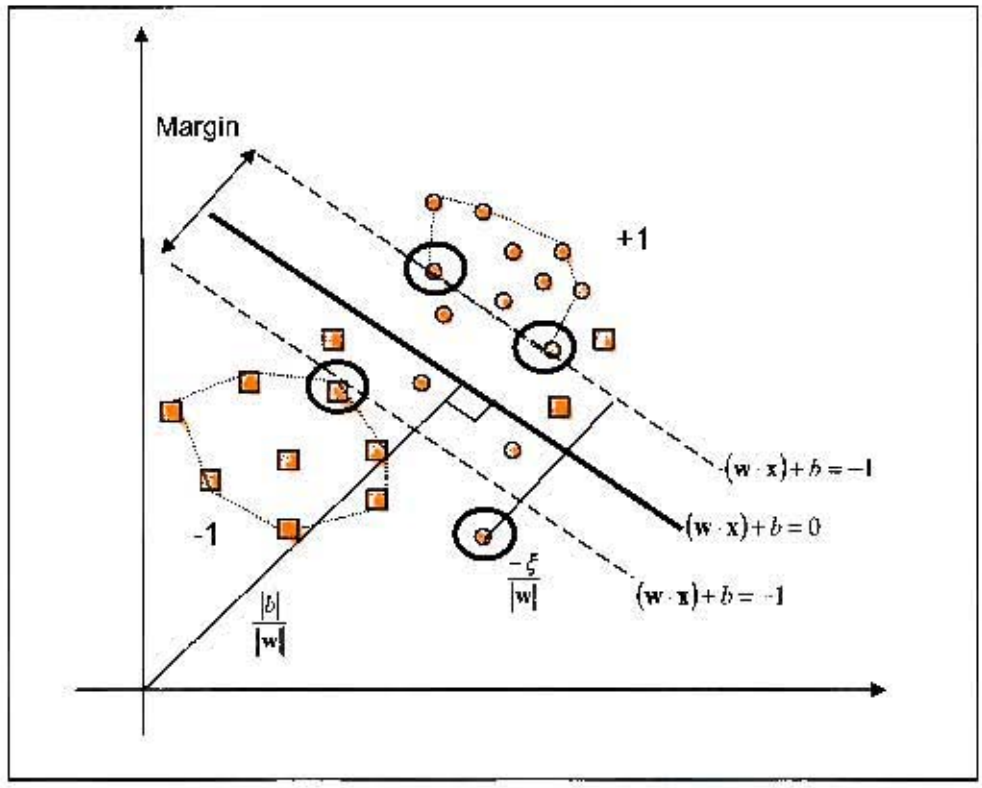

Figure 3.3: Separating plane for non-separable data[4].

Applying similar principles as applied to the linearly separable case we introduce the Lagragian

$$L=\frac{1}{2}\|\mathbf{w}\|^2+C\sum_i\xi_i-\sum_i\alpha_i\{y_i(\mathbf{x}_i\cdot\mathbf{w}+b)-1+\xi_i\}-\sum_i\mu_i\xi_i \qquad (3.34)$$

along with Lagrange multipliers μ_i which enforce the positivity of the slack variables.

The KKT conditions then become

$$\frac{\partial}{\partial w_k}L=w_k-\sum_i\alpha_i y_i x_{ik}=0;\quad k=1,\ldots,d \qquad (3.35)$$

$$\frac{\partial}{\partial b}L=-\sum_i\alpha_i y_i=0 \qquad (3.36)$$

[4] The circled points are the support vectors.

$$\frac{\partial}{\partial \xi_i} = C - \alpha_i - \mu_i = 0 \tag{3.37}$$

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i \geq 0; i = 1,\ldots,l \tag{3.38}$$

$$\xi_i \geq 0 \qquad \forall i \tag{3.39}$$

$$\alpha_i \geq 0 \qquad \forall i \tag{3.40}$$

$$\mu_i \geq 0 \qquad \forall i \tag{3.41}$$

$$\alpha_i\{y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i\} = 0 \tag{3.42}$$

$$\mu_i \xi_i = 0. \tag{3.43}$$

The KKT complementarity conditions are now (3.40) and (3.41). These are used, as before, to determine b. The set of support vectors now also includes those training points which are misclassified by the decision function.

3.4. Non-Linear Decision Boundaries

We have thus far only discussed the application of support vector classification to the linear problem. This section discusses the extension of the linear algorithm to applying non-linear decision boundaries. Since the data only appear in the training problem in the form of dot products $\mathbf{x}_i \cdot \mathbf{x}_j$, the linear algorithm is easily generalized to the non-linear problem by applying the "Kernel trick".

Recall that in support vector classification the input vectors $\mathbf{x}$ are mapped into high-dimensional feature space H through some nonlinear mapping,

$$\Phi : \mathbf{R}^N \rightarrow \mathbf{H} \tag{3.44}$$

where the optimal separating hyperplane is constructed. This results in a nonlinear decision boundary in the input space. In [47] it was observed that for constructing the separating hyperplane in the feature space H, the space defined by H does not need to be

considered in explicit form, we only need to be able to calculate the inner products between the support vectors and the vectors in the feature space [9].

Having performed the mapping Φ, the SVM algorithm only depends on the data through dot product functions of the form $\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$. Let K be a function such that

$$\mathbf{K}(\mathbf{x}_i\cdot\mathbf{x}_j)=\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j) \tag{3.45}$$

The function K is commonly referred to as a *kernel* function. The training algorithm now only needs to use K without having to explicitly know what Φ is. The kernel function is required to satisfy a set of restrictions called the *Mercer's condition*. Some examples of common kernel function are shown in Table 3.1.

Table 3.1: Common kernel functions

Polynomial	$\mathbf{K}(\mathbf{x},\mathbf{y})=(\mathbf{x}\cdot\mathbf{y}+1)^p$
Gaussian Radial Basis Function	$\mathbf{K}(\mathbf{x},\mathbf{y})=e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$
Sigmodial Function	$\mathbf{K}(\mathbf{x},\mathbf{y})=\tanh(\kappa\mathbf{x}\cdot\mathbf{y}-\delta)$

The kernel function makes it possible to use very high, possibly infinite, dimensional spaces without increasing the complexity of the training algorithm. By replacing $\mathbf{x}_i\cdot\mathbf{x}_j$ with the kernel function, the algorithm is able to produce a solution in the high dimensional space in roughly the same amount of time it would take to train un-mapped data [19].

Using kernels that satisfy Mercer's condition, a nonlinear decision function

$$f(\mathbf{x})=\operatorname{sgn}\left(\sum_{i=1}^{ls}y_i\alpha_i\Phi(\mathbf{s}_i)\cdot\Phi(\mathbf{x})+b\right) \tag{3.46}$$

$$=\operatorname{sgn}\left(\sum_{i=1}^{ls}y_i\alpha_i\mathbf{K}(\mathbf{s}_i,\mathbf{x})+b\right) \tag{3.47}$$

can be constructed, where s_i are the support vectors. This is equivalent to a linear decision function in the feature space $\boldsymbol{H}$ [45]. A simplified graphical representation of the kernel trick applied is shown in Figure 3.4. In this scenario, the data are non-linearly separable in the input space $\boldsymbol{X}$. The data are projected onto a higher dimensional feature space $\boldsymbol{H}$ using a function $\boldsymbol{\Phi}$. This transformation results in the data being linearly separable in the higher dimensional feature space $\boldsymbol{H}$.

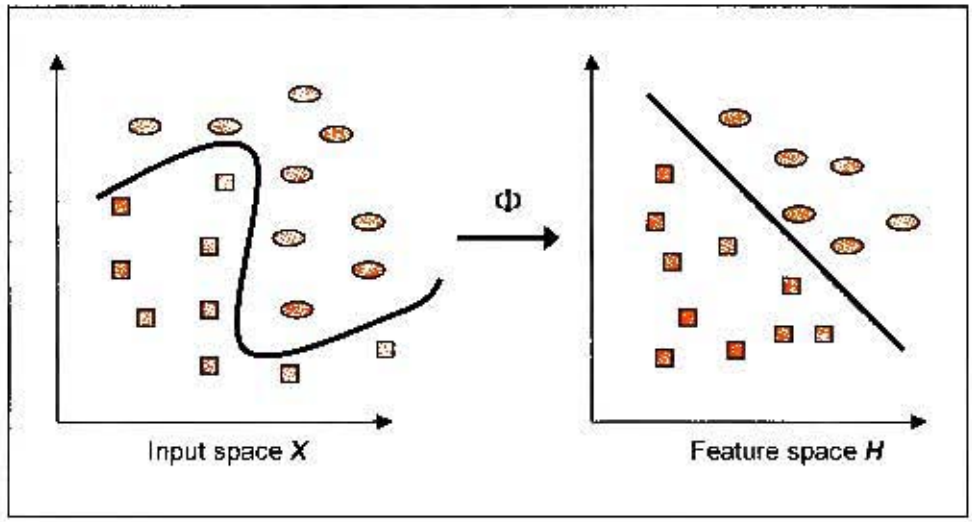

Figure 3.4: A simplistic representation of the kernel trick[5].

The introduction of kernels increases the expressive power of the learning machines while retaining the underlying linearity that will ensure that learning remains tractable [44].

3.5. Limitations of SVM

A major limitation for the SVM is its speed and size, primarily the extremely long training time [19]. Although the problem of speed has largely been solved for the test phase, training times, particularly for large datasets, remain problematic [19] . In fact, in [50] it was estimated that six years would be required to train an SVM on the data provided in the Timit database which consists of 630 speakers.

[5] Note that the feature space $\boldsymbol{H}$ is of a higher dimension than the input space $\boldsymbol{X}$.

Another limitation of the SVM is the choice of kernel. Once the kernel has been chosen, the only other user-chosen parameter is the error penalty. Although it has been argued that the choice of kernel does not make a great difference to the empirical performance "...the quality of decision rules obtained does not strongly depend on the type of SV machine..." [9]), there does remain the issue of the choice of parameters for the kernel function [51]. This is an important issue in optimizing SVM performance and is a valid research topic of its own [52, 53]. The best choice of kernel for a particular problem is still a research issue [19].

Also, although several solutions have been proposed, the SVM is fundamentally a binary classifier and the optimal design for multiclass SVM classifiers is an active area of research [19].

3.6. Related Work

Support vector machines have been successfully applied to several areas of pattern recognition such as off-line signature verification [54], text-categorization [55] and hand-written digit recognition [56]. This section will review some of the literature regarding the application of SVMs to speech related research. The review presented is by no means exhaustive.

The application of SVMs to classification tasks that are not binary classification involves further challenges because the support vector machine is by nature a binary classifier. However, the classifier has been successfully extended to the multi-class tasks using schemes such as one-vs.-all classifications. Much interesting research has been done in the effort of improving SVM performance in these tasks.

Schmidt [57] applied the support vector machine to a speaker identification task. The task in this research was a relatively difficult one using 10-15 seconds training speech and roughly five seconds test speech for each speaker. The task was also text-independent and conducted using speech collected over noisy telephone lines, which added to the

difficulty of identification. The results obtained by the SVM were compared to the results of a Bayesian classifier and it was found that the support vector system performed as well as, and sometimes slightly better than, the Bayesian classifier.

In Chin's work [58], SVMs were applied to a vowel recognition task and a speaker identification task. The results which are reported are promising, but what is more interesting is the detailed discussion of multi-class SVM classification and SVM parameter tuning that is included.

The Fisher kernel which was proposed by Jaakkola an Haussler [59] has been successfully applied in several examples of speech research. In his work presented in [10], Wan applied the SVM was to a speaker verification task using the technique of score-space kernels, including the Fisher kernel. This work investigated the differences in the performance of frame-level discrimination by a polynomial kernel and sequence-level discrimination using the Fisher and score-space kernels. The score-space kernels allow the entire speech sequence to be represented as a single vector, thus enabling the SVM to discriminate the whole sequence directly, as opposed to frame-by-frame discrimination. The score spaces approach is a kernel function that enables SVMs to classify whole sequences by explicitly mapping variable length sequence input vectors onto a single point in a fixed-dimension space called the score-space.

It was found that the SVM, combined with a score-space kernel and spherical normalization, out-performed a baseline Gaussian mixture model likelihood ratio approach on a speaker verification task performed on the PolyVar telephone database. Spherical normalization is a preconditioning technique that rescales the kernels preventing them from yielding large values.

Campbell [60] proposed a sequence kernel based on generalized linear discriminants and applied this technique to a text-independent speaker recognition task. The performance of the SVM classifier with this kernel was found to be competitive with results from the 1998 National Institute of Standards and Technology (NIST) Speaker Recognition

Evaluation (SRE). The NIST evaluations are held annually to showcase the work of researchers from academic institutions as well as industry and aim to *drive the technology forward, measure the state-of-the-art and find the most promising algorithmic approaches* [61].

Other work with SVMs in speech research has focused in developing hybrid systems that combine SVM with generative models such as Gaussian Mixture Models [14, 62] and Hidden Markov Models [63] .

3.7. Chapter Summary

This chapter provided an introduction to the formulation of Support Vector Machines for classification. The concepts of risk minimization, which are key to SVM formulation, were discussed. A review of some of the limitations of SVM classification as well as a discussion of some related literature is also included.

In the chapter that follows, feature normalization techniques and their limitations will be discussed.

CHAPTER FOUR

4. FEATURE NORMALIZATION

The two main sources of variation on the telephone channel are additive noise and a changing channel frequency response from telephone line to telephone line [64]. Short-term spectral features are highly sensitive to noise, the effects of which are somewhat complex [65].

In designing methods to improve the robustness of speech processing systems against channel effects and background noise a distinction is often made between *additive* and *convolutional* noise [15]. Robust speech techniques attempt to maintain the performance of speech processing systems under diverse conditions [16].

Feature-based normalization techniques aim to make the features generated in the parameterization process more robust to mismatched conditions [18]. Considering the time domain addition of noise $n(t)$ to the speech signal $s(t)$, the composite signal can be represented as [65]:

$$x(t) = s(t) + n(t). \tag{4.1}$$

The effect of the noise in the log power spectra is then represented by:

$$\log\left(x^{e^{jw}}\right)=\log\left(s^{e^{jw}}+n^{e^{jw}}\right). \tag{4.2}$$

In [3], it was observed that a shift in the mean and a reduction in variance were some of the effects that noise has on the distribution of the cepstral features.

Cepstral distributions for clean data are well behaved and approximately normal but take on a significantly different profile in the presence of noise [65]. Feature normalization techniques therefore tend to aim to normalize the cepstral distributions in the presence of noise.

This chapter gives a brief review of certain feature normalization techniques which have been applied to speech processing tasks. The techniques which are discussed are Cepstral Mean Subtraction, Mean Variance Normalization, RASTA Speech Processing and Histogram Normalization.

4.1 Cepstral Mean Normalization

Speech signals that are transmitted over a telephone network often encounter a linear distortion due to the filtering effect of the channel [16]. This filtering effect is expressed as [16]:

$$S_F(z)=S(z)T(z) \tag{4.3}$$

where $S(z)$ corresponds to the original speech, $T(z)$ corresponds to the telephone channel and $S_T(z)$ corresponds to the filtered speech [16]. In the logarithmic domain this corresponds to [16]:

$$\log S_F(z)=\log S(z)+\log T(z) \tag{4.4}$$

The cepstrum of the filtered signal is thus equal to the cepstrum of the speech signal added to the cepstrum of the filter such that [18]:

$$c_{S_F} = c_S + c_T \tag{4.5}$$

The linear filter effect of different transmission channels is an example of convolutional noise [15]. It is assumed that this linear filtering effect is also time-invariant [18]. In the cepstral domain, convolutional noise causes a constant offset in the long-term average of the cepstral coefficients [15]. Cepstral Mean Normalization (CMN) or Cepstral Mean Subtraction (CMS) is used to compensate for the effect of the telephone channel on the speech. This is performed by subtracting the long-term average from the cepstral coefficients on a per-utterance basis [66].

It is also assumed that the duration of the speech is long enough such that the average speech spectrum is flat [18]. This compensation technique has the effect of causing the mean of the distribution of the compensated feature values to be equal to zero which normalizes the first moment of the distribution [18].

An important advantage of removing the time averages is that all time-invariant frequency distortions in the data are eliminated [35].

Other than the filter effect, CMN has also been found to compensate for some of the effects of inter-session variability [18]. CMN can be interpreted as a high-pass filter as it effectively removes the DC component of the data [18]. CMN makes the mean of the compensated variable zero and so equalizes the first moment of its probability distribution [67].

There have been extensions made to CMN such as *SNR-Dependent Cepstral Normalization* (SDCN) which applies an additive correction in the cepstral domain that depends exclusively on the instantaneous SNR of the signal [68]. Although the SDCN algorithm is simple and effective it requires environment-specific training [68]. Other

extensions to CMN are *Codeword-Dependent Cepstral Normalization* (CDCN) and *Phone-Dependent Cepstral Normalization* (PDCN) [69].

4.1.1 Limitations of CMN

The effectiveness of CMN is limited by its inability to compensate for nonlinear effects of additive noise and telephone transmission [18]. It also tends to degrade system performance when training and testing are done in matching channel conditions [16, 18]. This is largely due to the assumption that the long term cepstral mean of the clean speech is zero. This requires that the speech segment include approximately equal amounts of voiced, unvoiced and plosive sounds [16, 18]. Also, the long-term average required by CMN may be difficult to obtain in real-time implementations [17].

4.2 Mean Variance Normalization

Mean Variance Normalization (MVN) is a natural extension of CMN which normalizes the first two moments of the probability distribution by applying the linear transformation [70]:

$$x' = \frac{x - \mu}{\sigma} \tag{4.6}$$

where μ is the long term mean and σ is the variance.

Additive noise has been shown to result is a reduction of the variance of the cepstral features [65]. Normalizing the variance has been found to improve the speaker recognition system performance. In MVN the mean and the variance of the cepstral coefficients of clean speech are assumed to be invariant [70].

4.2.1 Limitations of MVN

Similarly to CMN, MVN is only able to compensate for linear frequency distortions to the data. MVN is only able to provide compensation for the first two moments of the probability distribution of the speech features [71].

4.3 RASTA Speech Processing

Relative Spectral speech processing is another feature-based compensation technique which is used to improve the performance of speech processing systems in adverse conditions [18]. RASTA processing takes advantage of the fact that the rate of change on the non-linguistic components in a speech signal often lies outside the typical rate of change of the vocal tract shape (which is the component of interest) [22]. This is achieved by suppressing those spectral components that change more slowly or quickly than the typical rate of change of speech [22]. So in the same way that CMN can be interpreted as a high-pass filter, RASTA processing can be interpreted as a band-pass filter.

In RASTA processing the constant factors in each spectral component of the spectrum are suppressed by passing the running spectral estimate through a band-pass or high-pass filter with a sharp spectral zero at zero frequency [2].

Human hearing is relatively insensitive to slowly varying stimuli (such as the slow changing frequency characteristics of communication channels as well as steady background noise). RASTA speech processing aims to apply this logic to machine speech processing [22].

Similarly to CMN the RASTA technique attempts to make the long-term average log spectrum zero [22].

As stated by Hermansky [22], "*the main difference between cepstral mean subtraction and the RASTA processing in the cepstral domain is that the CMS merely removes the dc component of the short term log spectrum whereas the RASTA processing influences the speech in a more complex way...*"

4.3.1 Limitations of RASTA Speech Processing

The RASTA method compensates primarily for differences in the frequency response of the channel and is not able to deal with the combined effects of additive noise and filtering by the channel [2]. Also, part of the speech spectrum is also removed in the filtering operation, particularly if the frame energy component of the cepstral vector is included in the filtering operation.

4.4 Histogram Normalization

Although linear feature-based compensation techniques like CMN and MVN have been shown to improve the speaker verification performance in adverse conditions, they are unable to compensate for non-linear distortions of the feature space [72]. HN operates under the assumption that the shape of the entire distribution of the cepstral coefficients of clean speech is invariant and any detail of the cepstral distribution is regarded irrelevant and to be removed [70].

The basis of HN, also referred to as Histogram Equalization (HEQ), is that the distribution of the cepstral coefficients in the test data should be identical to that of the training data [70]. For the remainder of this thesis the terms "histogram normalization" and "histogram equalization" will be used interchangeably.

The aim of HEQ is to provide a transformation $x(y)$ which converts the probability distribution $p_y(y)$ of the noisy speech into a reference distribution $p_x(x)$ corresponding to

the clean speech [73]. It can be shown that if $x(y)$ transforms $p_y(y)$ into $p_x(x)$, then the cumulative histograms verify that [73]:

$$C_y(y) = C_x(x(y)) \tag{4.7}$$

For a random variable y with probability density function $p_y(y)$, a function $x = F(y)$ which maps $p_y(y)$ into a reference distribution $p_x(x)$ can be obtained by equating the cumulative distribution function (CDF) of x and y [74]:

$$C_y(y) = C_x(x) = C_x(F(y)) \tag{4.8}$$

The transformation can be obtained from the cumulative histogram of the noisy speech and the reference cumulative histogram for the clean speech such that [73]:

$$x(y) = F(y) = C_x^{-1}[C_y(y)] \tag{4.9}$$

In general, $F(y)$ is monotonic non-decreasing and nonlinear [74]. Under the assumption of statistical independence, HEQ is applied to each cepstral coefficient independently [74]. For each input utterance, the CDF of each coefficient is approximated by its cumulative histogram [74].

The practical implementation of HEQ would proceed as follows [18, 75]:

(i) Find the minimum and maximum values x_{min} and x_{max}, across the whole data set, for each dimension.

(ii) Divide the range $[x_{min}, x_{max}]$ into N equally spaced, non-overlapping intervals such that $x_{\min} = b_1 < b_2 < ...b_{N+1}$ and the bin B_i is defined by $B_i = [b_i, b_{i+1})$.

(iii) Use these bins to construct a histogram across the entire data set. This is achieved by scanning each dimension of the data set and determining the number of observations that fall into each of the bins.

(iv) Approximate the probability of observation x being in bin B_i by $p_x(x \in B_i) = \frac{n_i}{N_x}$ where n_i is the number of observations in bin B_i and N_x is the total number of observations in the set. This probability approximation is simply a normalised version of the histogram obtained in (iii) above.

(v) The cumulative distribution function is then approximated by $C_x(x : x \in B_i) = \sum_{j=1}^{i} \frac{n_j}{N_x}$. This cumulative histogram is a piece-wise constant function approximation of the true cumulative distribution function.

(vi) Each value of x is then replaced by the value of y that corresponds to the same point in the reference and computed cumulative histograms such that $C_x(x) = C_{ref}(y)$.

Step (vi) is implemented efficiently by constructing two lookup tables $\{x, C_x\}$ and $\{y, C_{ref}\}$ from $C_x(x)$ and $C_{ref}(y)$ respectively such that C_x and C_{ref} take on values in the range [0, 1] in equal increments. These tables are then combined to give a new table $\{x, y\}$ which is a piecewise constant approximation of the true transformation function [18, 75]. Then for every value of x, the closest value of y can be found using a binary search and used as the transformed value of x.

HEQ based transformation can be viewed as an extension of linear transformation techniques like CMN and MVN that only deal with the normalization of the two first moments of the probability distributions of the features [74].

4.4.1 Limitations of Histogram Normalization

A drawback of HEQ based techniques is that the nonlinear transformation is based on mapping the global CDF of each feature into a reference one [74]. When the CDF estimation is built using a reduced number of observations from a single utterance,

variations in the amount of non-speech frames in the utterance introduces unwanted variability in the estimated CDF and in the corresponding transformation function [74]. This variability may lead to degradation in the performance of the technique. This can be avoided by using a voice activity detector to discard non-speech frames prior to applying HEQ [74].

4.5 Related Work

Improving the performance of speech processing systems in adverse conditions is an on-going challenge and the development of feature normalization techniques has played a big part in this. This section discusses some of the relevant literature relating to the development and application of feature normalization techniques to speech related applications.

Cepstral Mean Normalization is arguably the most commonly used method of feature-based noise compensation. In [16] it was found that removing the mean of the cepstral feature vector significantly improved the system accuracy where training and testing were done on mismatched channels.

Histogram Equalization has its roots in image processing. However, there has been an increasing interest in the application of this technique to speech related research [18, 67, 70, 71, 73, 75]. All the literature is in agreement that HEQ and HEQ-based techniques outperform CMN and MVN because the latter are linear normalization techniques and fail to adequately compensate for non-linear distortions that might be introduced to the speech signal. In his work in [18], Skosan provides a comprehensive review of literature relating to HEQ and its application to speech research.

In [18, 72] HEQ was applied to the speaker verification task and found to improve the performance of a GMM-based speaker verification system. The performance of a segmental version of the algorithm was compared to that of the classical HEQ method. In

the modified segmental HEQ (MSHEQ) algorithm the normalization procedure is applied to features extracted from short adjacent segments of speech within an utterance. This differs from non-segmental HEQ which is applied on an utterance-by-utterance basis. The MSHEQ algorithm was found to outperform its non-segmental counterpart.

4.6 Chapter Summary

In this chapter some common feature normalization techniques and the limitations of each were discussed. A review of literature relating to the application of feature normalization techniques in speech related fields is also given.

The next chapter discusses feature extraction techniques PCA and ICA.

CHAPTER FIVE

5. FEATURE EXTRACTION

In feature extraction the parameter/feature vector is transformed to a vector y which has dimensionality less than or equal to, that of the parameter vector. These transformation methods may have the effect of improving classification as well as reducing the data dimensionality by search for transformation that emphasize more important features while suppressing less desirable ones [76].

Feature extraction is applied with the aim of reducing the computational cost [20]. The main aim of the feature extraction step is to *select or combine the features that preserve most of the information and remove the redundant components in order to improve the efficiency of the classifiers without degrading their performances* [20]. Thus the goal of these algorithms is to obtain a compact, accurate representation of the data that reduces or eliminates statistically redundant components [77]

Linear feature extraction generates feature vectors by projecting parameter vector onto a feature space through a linear transformation $\mathbf{T}_{pxm}, p \geq m$ such that [20]:

$$y = \mathbf{T}^T x \tag{5.1}$$

where x is the parameter vector and y is the transformed vector. The primary difference between linear feature extraction algorithms is the optimization criteria that are used in constructing the transformation matrix.

Feature extraction can be performed simultaneously with the classification step or it may be performed independently. This chapter discusses two linear independent feature extraction techniques namely Principal Component Analysis and Independent Component Analysis. The application of these techniques to speaker recognition tasks is also discussed.

5.1. Principal Component Analysis

Principal Component Analysis (PCA) is a well established feature extraction technique, first introduced in 1901 [20, 78]. In PCA the transformation matrix $\boldsymbol{T}$ is obtained by searching for the directions that have the largest variation [20]. The central idea of PCA is to reduce the dimensionality of a dataset that contains a large number of interrelated variables while retaining as much of the variation present in the data, as possible [20]. This as achieved by transforming the data set to a new set of variables; the principal components, which are uncorrelated and ordered such that the first few retain most of the variation present in the original dataset [78].

Principal Component Analysis represents a linear transformation where the data are expressed in a new coordinate basis that corresponds to the direction of maximum variance [79]. The idea is that this new basis will filter out the noise in the data and reveal hidden structure [80]. An advantage of PCA is that once these hidden structures have been found, the data can be compressed (by reducing the number of dimensions) with minimal loss of information [81]. The PCA transformation not only decorrelates the original signal components but the projection along the directions of highest variance also maximizes the variance and minimizes the average squared residual between the original

signal and its dimension-reduced approximation [77]. The technique is widely used in face and image recognition.

Let X represent a dataset consisting of N observation vectors $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}, x_i \in R^d$. It is assumed that PCA behaves very sensitively when the magnitudes of the observation vector components are significantly different. The observation vectors are therefore typically standardized to zero mean and unit variance. The corresponding sample covariance matrix is given by:

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T \tag{5.2}$$

The principal components are computed by solving the eigenvalue problem

$$\lambda \mathbf{v} = \mathbf{C}\mathbf{v} \tag{5.3}$$

where $\lambda \geq 0$.

Generally, the m principal components are given by the m leading eigenvectors of the sample covariance variance. That is, those eigenvectors corresponding to the m largest eigenvalues. These eigenvectors are used as the columns of the transformation matrix T. The transformed data are then given by $y = \mathbf{T}^T x$.

Generally in PCA the transformation matrix vectors are chosen such that $m < n$. A graphical representation of the effect of PCA is shown in Figure 5.1. Through the PCA transformation the data are transformed onto the directions of maximum variance (x_2,y_2).

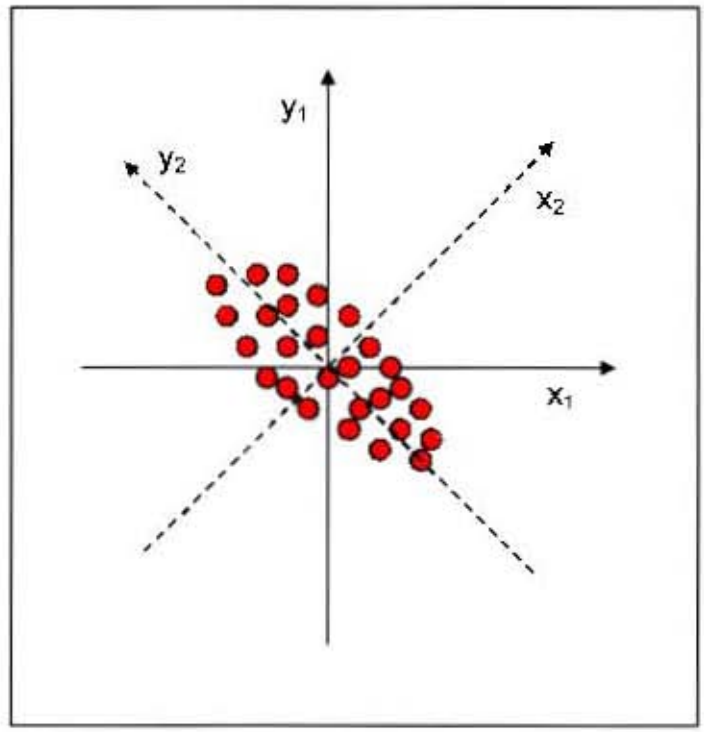

Figure 5.1: A simplified graphical representation of PCA.

There are some important benefits to be gained from dimensionality reduction by PCA. First, there is the possibility of reducing computational overhead in subsequent processing stages. Also, the noise may be reduced, assuming that the data not contained in the first m contain mostly noise. Finally, a projection onto a very low dimension subspace can be useful for visualization of the data [82].

5.1.1. Limitations of PCA

A possible shortcoming of PCA, particularly in classification applications, is that the projection of data onto the maximally variant directions does not necessarily translate to maximising the discrimination between the data classes. Also PCA is scale-sensitive which makes the transformation vulnerable to being dominated by elements with particularly large variance [20]. If there are large differences between the variances of the elements of the original dataset then those variables with the largest variance will tend to

dominate the first few principal components [78]. Another limitation of PCA is that the extracted features are not invariant under transformation and even scaling the attributes changes the resultant features [83]. Also, although uncorrelated, the principal components can be highly statistically dependent which can lead to PCA failing to find the most compact representation of the data [77].

5.2. Independent Component Analysis

Independent Component Analysis (ICA) is a statistical signal processing technique the goal of which is to express a set of random variables as linear combinations of statistically independent component variables [84]. The difference in the optimization criterion of the transformation matrices of PCA and ICA is that the ICA components are maximally statistically independent while the PCA components which are maximally variant.

ICA is very often associated with the *cocktail party* problem. Consider a scenario where there are n people speaking in a room with m microphones. Each of the microphones detects a signal which is a mixture of the n individual source signals. The individual voices of the people in the room are independent source signals and the signals detected by each of the microphones are linear mixtures of these source signals. Through the ICA process we are able to separate these mixtures into the source signals without needing much information about the sources. In fact, ICA requires only the assumption that the source signals are statistically independent and have nongaussian distributions.

The concept of statistical independence can be explained as follows. Let $x_1,...,x_m$ be random, centred variables. The variables are centred by subtracting the mean of the vector [82]. The variables x_i are mutually independent if the joint density function can be factorised to [82]:

$$f(x_1,...,x_2) = f_1 x_1,...,f_m x_m \tag{5.4}$$

where $f_i x_i$ is the marginal density of x_i. ICA exploits the fact that two signals, such as voices, from different physical sources are independent [85].

A general definition for ICA is given by [82]: *The ICA of a random vector* $\mathbf{x}$, $\mathbf{x} = (x_1,...,x_n)^T$, *consists of finding a linear transform* $\mathbf{s} = \mathbf{W}\mathbf{x}$ *so that the components* s_i *are as independent as possible in the sense of maximizing some function* $F(s_1,...,s_k)$ *that is a measure of independence.*

In the simplest form of ICA, n scalar random variables $x_1,...,x_n$ are observed. These are assumed to be linear combinations of k unknown independent components $s_1,...,s_k$ that are mutually statistically independent and zero-mean [84].

An alternative ICA definition is given by [82]: *The ICA of a random vector* $\mathbf{x}$, $\mathbf{x} = (x_1,...,x_n)^T$, *consists of estimating the following generative model for the data*

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{5.5}$$

where the components s_i *in the vector* $\mathbf{s} = (s_1,...,s_k)^T$ *are assumed to be independent. The matrix* $\mathbf{A}$ *is a constant* $n \times k$ mixing *matrix.*

Denoting the columns of $\mathbf{A}$ as $\mathbf{a}_j$, the ICA model can also be expressed as [86]:

$$\mathbf{x} = \sum_{i=1}^{k} \mathbf{a}_i s_i \,. \tag{5.6}$$

The ICA model is a generative model, which means that it describes how the observed data are generated by a process of mixing the components s_i [86].

The ICA process is represented graphically in the figure 5.2.

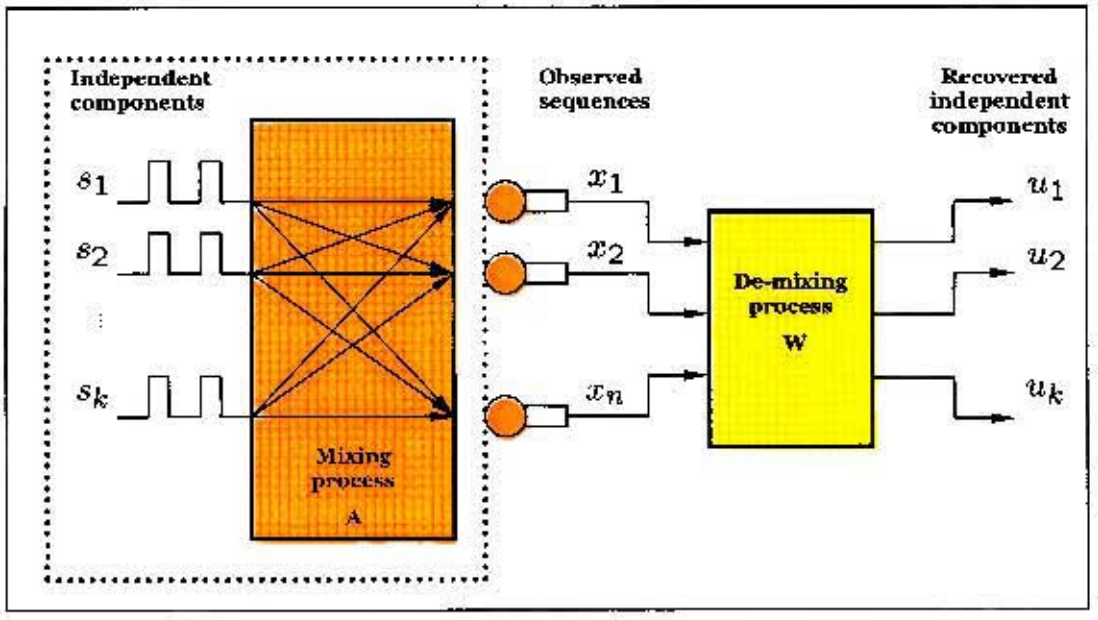

Figure 5.2: Graphical representation of independent component analysis[iii].

The starting point of ICA is the assumption that the components are statistically independent [86]. Thereafter, the following restrictions must be met in order to successfully calculate the ICA model [82]:

(i) All the independent components s_i with the possible exception of one component, must have a nongaussian distribution

(ii) The number of observed linear mixtures n must be at least as large as the number of components k ($n \geq k$)

(iii) The matrix **A** must be of full column rank.

Once the mixing matrix **A** has been calculated, its inverse $\mathbf{W} = \mathbf{A}^{-1}$ is used to compute the independent components by:

$$\mathbf{s} = \mathbf{W}\mathbf{x}. \tag{5.7}$$

There are certain ambiguities within the ICA model that deserve mention [82, 86]:

- The variance of the independent components cannot be determined.

The columns of **A** and the independent components can only be estimated up to a multiplicative constant. A scalar multiplier in one of the sources s_i can be cancelled by dividing the corresponding column $\mathbf{a}_i$ of **A** by the same scalar. For mathematical convenience the independent components are thus generally defined to have unit variance.

- It is generally not possible to determine an order for the independent components.

Since both **s** and **A** are unknown, the order of the terms in the summation $\mathbf{x} = \sum_{i=1}^{n} \mathbf{a}_i s_i$ can be freely changed and any of the independent components could be the 'first' one. This is in contrast to PCA where the components are ordered by the magnitudes of the corresponding eigenvalues.

A possible method for introducing an order to the independent components would be to use the norms of the columns of the mixing matrix **A**. This norm gives the contributions of the independent components to the variances of x_i and ordering the components according to descending norm of the corresponding columns of the mixing matrix results in an order that is reminiscent of PCA [82]. Another approach would be to measure the nongaussianity of the independent components and order them according to this measure [82].

Though ICA has several possible applications, this thesis is concerned only with the application of ICA to feature extraction. In this application the columns of **A** represent the features and s_i is the coefficient of the i-th feature in an observed data vector **x** [82]. Similarly to PCA, ICA can be used to reduce the data dimensionality by reducing the number of components included in the transformation matrix.

As mentioned, nongaussianity is a key requirement in the computation of the ICA model. The following section briefly describes methods that are used to measure the nongaussianity of a random vector.

5.2.1. Measuring Nongaussianity

Nongaussianity is imperative to estimating the ICA model. The estimation is, in fact, not possible without the nongaussianity restriction [82, 86]. In order to use nongaussianity in the ICA model estimation, a quantitative measure of the nongaussianity of a random variable needs to be defined.

5.2.1.1. Kurtosis

Kurtosis is a classical measure of nongaussianity. The kurtosis of a random variable y is defined by [86]:

$$kurt(y) = E\{y^4\} - 3(E\{y^2\})^2 \tag{5.8}$$

The random variable y is assumed to be of unit variance which simplifies (5.4) to

$$kurt(y) = E\{y^4\} - 3 \tag{5.9}$$

The kurtosis is thus simply a normalized version of the fourth moment $E\{y^4\}$. It can be shown that the kurtosis for a gaussian random variable will be zero while nongaussian variables typically have non-zero kurtosis [86].

The kurtosis value can be negative or positive. As such, nongaussianity is generally measured by the absolute value or the square of the kurtosis. These measures are zero for gaussian variables and greater than zero for most nongaussian random variables [86].

An advantage of kurtosis as a measure of gaussianity is the computational and theoretical simplicity. However, when applied in practice, it suffers the drawback of being very sensitive to outliers in the observed samples and is not a robust measurement of nongaussianity [86].

5.2.1.2. Negentropy

Negentropy, which is based on the information theoretic quantity of differential entropy, can also be used as a measure of nongaussianity [86, 88]. The entropy of a given random variable can be interpreted as the degree of information that is obtained from the observation of the random variable. As the unpredictability and 'randomness' of the variable increases, so does its entropy [86]. A fundamental result of information theory is that a gaussian variable has the largest entropy among all variables of equal variance [86]. Entropy can thus be applied as a measure of gaussianity.

The entropy H of a discrete random variable y is given by:

$$H(\mathbf{y}) = -\sum_i P(\mathbf{y} = a_i) \log P(\mathbf{y} = a_i) \tag{5.10}$$

where a_i are the possible values of y.

This definition generalises to the differential entropy of a random vector $\mathbf{y} = (y_1, ..., y_n)^T$ given by:

$$H(\mathbf{y}) = -\int f(\mathbf{y}) \log f(\mathbf{y}) dy \tag{5.11}$$

where $f(\mathbf{y})$ is the density of $\mathbf{y}$.

The negentropy of $\mathbf{y}$ is then defined as:

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{5.12}$$

where $\mathbf{y}_{gauss}$ is a gaussian random vector of the same covariance matrix as **y**. Negentropy is always non-negative and is only ever zero if and only if **y** has a gaussian distribution. It is also invariant for invertible linear transformations.

The advantage of using negentropy as a measure of nongaussianity is that it has a solid foundation in statistical theory and is in some sense the optimal estimator of nongaussianity. It is, however, a computationally complex task and in practise approximations of negentropy are used as opposed to the formal definition in (5.12).

Hyvärinen developed approximations of negentropy based on the maximum-entropy principle [86]. The approximation is generally given by:

$$J(\mathbf{y}) \approx \sum_{i=1}^{p} k_i [E\{G_i(\mathbf{y})\} - E\{G_i(v)\}]^2 \tag{5.13}$$

where k_i are some positive constants; v is a standardized gaussian variable (zero mean, unit variance) and G_i are some nonquadratic functions. In the case where only one nonquadratic function is needed (5.13) becomes

$$J(\mathbf{y}) \propto [E\{G(\mathbf{y})\} - E\{G(v)\}]^2 \tag{5.14}$$

The negentropy approximation then depends on the choice of the function G. Some examples of functions that are commonly used are shown in Table 5.1.

Table 5.1: Examples of contrast functions for negentropy approximation.

$G_1(u) = \log\cosh(u)$
$G_2(u) = -\exp\left(\frac{1}{2}u^2\right)$
$G_3(u) = \frac{1}{4}u^4$

Note that using the last nonlinearity $G(u)=\frac{1}{4}u^4$ one obtains a kurtosis based approximation.

5.2.1.3. Mutual Information

The mutual information I between m scalar random variables $y_i, i=1,...,m$ is defined as [87, 89]

$$I(y_1, y_2, ..., y_m)=\sum_{i=1}^{m} H(y_i)-H(\mathbf{y}) \tag{5.15}$$

It is a natural measure of dependence and can intuitively be understood as measuring the amount of information the variables y_i give about each other. If y_i are independent they give no information about each other. Mutual information is always non-negative and is zero if and only if the random variables are statistically independent.

For an invertible, linear transformation $\mathbf{W}$ where $\mathbf{y} = \mathbf{W}\mathbf{x}$ then

$$I(y_1, y_2, ..., y_m)=\sum_{i} H(y_i)-H(\mathbf{x})-\log|\det \mathbf{W}|. \tag{5.16}$$

Let y_i be constrained to be uncorrelated and have unit variance then

$$E\{\mathbf{y}\mathbf{y}^T\}=\mathbf{W}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}^T=\mathbf{I} \tag{5.17}$$

which implies

$$\det \mathbf{I}=1=(\det \mathbf{W}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}^T)=(\det \mathbf{W})(\det E\{\mathbf{x}\mathbf{x}^T\})(\det \mathbf{W}^T) \tag{5.18}$$

which in turn implies that $\det\mathbf{W}$ is constant.

For unit variance y_i, entropy and negentropy differ only by a constant and sign, therefore

$$I(y_1, y_2, ..., y_m) = C - \sum_i J(y_i) \tag{5.19}$$

where C is a constant that does not depend on **W**.

From (5.19) it can be seen that finding an invertible transformation **W** that minimizes the mutual information is equivalent to finding the directions in which the negentropy is maximized [87, 89].

5.2.2. The FastICA Algorithm

The FastICA algorithm, developed by Hyvärinen [82, 86, 88, 90] is a widely used algorithm for performing independent component analysis. It is an efficient, fixed-point algorithm and can also be applied to projection pursuit. The algorithm is based in the contrast functions shown in Table 5.1.

A simplistic formulation of the algorithm is as follows:

Let g be the derivative of the nonquadratic function G in (5.14) be as shown in Table 5.2.

The FastICA algorithm then proceeds as [86]:

1. Choose an initial weight vector **w**.
2. Let $\mathbf{w}^+ = E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w}$
3. Let $\mathbf{w} = \mathbf{w}^+ / \|\mathbf{w}^+\|$
4. If not converged, go back to step 2.

Convergence in this case means that the dot product of the old and new **w** is (almost) equal to 1, i.e. the old and new **w** point in the same direction. The independent components can therefore only be defined up to the multiplicative sign.

Table 5.2: FastICA algorithm nonlinearities.

$g_1 = \tanh(u)$
$g_2(u) = u\exp\left(\frac{1}{2}u^2\right)$
$g_3(u) = u^3$

Some important properties of the algorithm are:

- The algorithm finds directly independent components of (practically) any nongaussian distribution using and any nonlinearity.
- The convergence of the algorithm is at least quadratic making it faster than algorithms which converge linearly.
- The algorithm performance can be optimised by choosing the appropriate nonlinearity g.
- It is parallel, distributed, and computationally simple and requires little memory space.

5.2.3. Limitations of ICA

The strong nongaussianity requirement of ICA has acted as a limiting factor to its application to speech processing. Without nongaussianity, the estimation of the ICA model is not possible. In most statistical theory, including speech processing applications, the random variables are assumed to have gaussian distributions thus precluding the use of any methods related to ICA [86]. However, there has recently been an increased interest in the application of ICA and related methods to speech related research [91-93]. Another limitation of ICA is that the ICA algorithm is iterative and convergence can sometimes be difficult [94].

5.3. Related Work

There are several examples in literature of the inclusion of feature extraction techniques to speech processing. This section gives a brief review of some that are relevant to the work that was carried out in this thesis.

In [95] ICA feature vector transformation was applied to a text-independent speaker identification task on telephone speech. The work compared the performance of the ICA transformation to a PCA transformation. The transformation was applied to the cepstrum vector and the transformed cepstrum vectors were used for training and test. Speaker identification used a vector quantization based system. No dimensionality reduction was applied and the results showed an improvement of up to 3% when using ICA in adverse environments. It was found that the ICA transformation outperformed the PCA transformation. The authors also compared the performance of the three contrast function listed in tables 5.1 and 5.2. They found that the kurtosis based approximation, on average, outperformed the other two.

In [96] a text-independent speaker recognition system which applied feature transformation was proposed. The performances of ICA and PCA feature transformation applied to mel-frequency cepstral coefficient features were compared. Gaussian Mixture Modelling was used for modelling the speakers and it was found that the ICA transformation outperformed the PCA transformation on both clean and noisy speech.

In [20] the Reduced-Dimensional SVM (RDSVM) algorithm was proposed by Wang. The basic idea of RDSVM is to reduce the computational burden of traditional SVM training by reducing the number of computations in the kernel functions. The foundation for this is as follows.

The basis of SVM is the mapping of the input space onto a high dimensional space by some mapping Φ (see Chapter Three for details). The objective function is ensured to be a convex programming problem by the use of a kernel function which is defined such that

$$\mathbf{K}(\mathbf{x}_i \cdot \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j). \tag{5.20}$$

Since the number of observation vectors can generally not be reduced to very low, Wang suggests that an effective way of reducing the number of kernel function computations is to reduce the dimensionality of the observation vectors.

RDSVM is a two layer structure where the first layer reduces the data dimensionality and the second layer conducts SVM training in the new, reduced-dimensional feature space. This is depicted graphically on Figure 5.2.

Wang applied the RDSVM algorithm to a speech recognition task using the TIMIT database. He found that RDSVM performed as well as, if not slightly better than, traditional SVM. He also found that RDSVM performance on testing data only started degrading when the feature dimension was less than five on speaker dependent speech recognition. The feature dimension was reduced from 21.

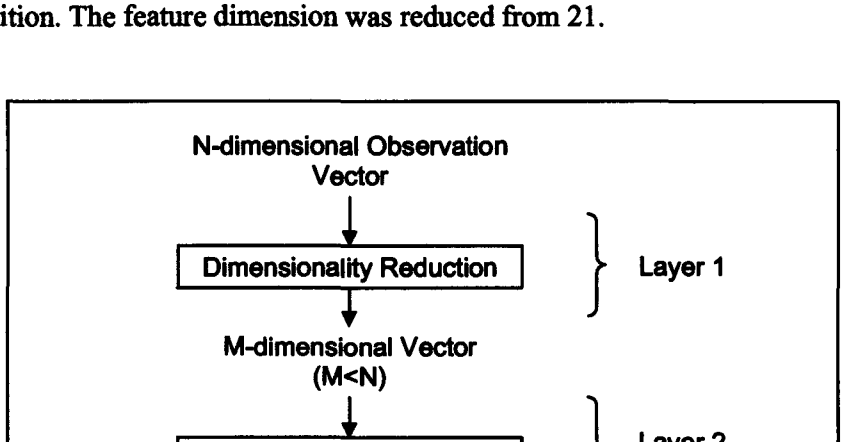

Figure 5.3: Reduced- Dimensional SVM Structure

In applying the RDSVM algorithm Wang used a Generalised Minimum Classification Error training algorithm with a linear discriminant initialization in Layer 1.

In [97] PCA and ICA were applied to feature extraction for SVM applied to a time-series forecast task. The linear feature extraction techniques were also compared to the Kernel PCA, a nonlinear form of PCA based on the kernel method described in section 3.4. The results in that work showed that SVM performed better with feature extraction as opposed to without. It was found that the generalization ability of SVM improved with the application of feature extraction. It was also concluded that ICA and KPCA outperformed PCA feature extraction.

The work presented in this thesis applies the RDSVM structure mentioned above to a speaker verification task using PCA and ICA feature extraction. This differs from [97] which was discussed above where the task was a regression as opposed to classification.

The combination of SVM and ICA has been successfully applied to other classification tasks such as face feature extraction [98] and face detection [99].

5.4. Chapter Summary

This chapter discussed feature extraction techniques PCA and ICA. These techniques can also be used to reduce data dimensionality. The limitations of these were also discussed. The chapter concluded with a review of some relevant literature.

In the following chapter the experimental framework for this research is reviewed.

CHAPTER SIX

6. EXPERIMENTAL FRAMEWORK

This chapter discusses the experimental procedure that was followed for the work presented in this thesis. A review of the speech corpora and software tools used is also given.

6.1 Objectives

The work presented in this thesis focuses on a text-independent speaker verification task using conversational telephone speech data. The work carried out in this project aimed to:

1. Establish the effect of histogram equalization on the performance of a support vector machine classifier when applied to a speaker verification task using telephonic speech data. There are examples of HEQ being successfully applied to other speech related tasks and used with other classifiers such as the gaussian mixture model [18, 67, 72-75]. It is expected that the application of HEQ will improve the accuracy of the SVM on this task. Also, the impact of HEQ will be

compared to that of the well established linear compensation techniques, Cepstral Mean Subtraction and Mean Variance Normalization.

2. Although the SVM has very good generalization capabilities, it has the distinct disadvantage of very long training times. In this project we apply the Reduced Dimensional SVM model proposed in [20] to the text-independent speaker verification task. Two linear, independent feature extraction techniques, namely Principal Component Analysis and Independent Component Analysis, are considered for this task.

The following sections discuss the corpus and software tools which were used for this project.

6.2 Experimental Corpus

The experiments were carried out using the National Institute of Standards and Technology (NIST) 2000 Speaker Recognition Evaluation (SRE). The speech data in this database was extracted from the Switchboard-II corpus, phases 1 and 2 collected by the Linguistic Data Consortium. It consists of conversational telephone speech.

There are 2521 female and 2470 male trials in the complete evaluation. However, due to the long training times of the SVM models, a subset of the female one-speaker detection trials was performed for this thesis. In one-speaker detection the task is to determine whether a specified speaker is speaking in a given segment of speech. The database provides both training and testing data. The training speech segments are typically two minutes in length while the test segments vary from 15 to 45 seconds and are sometimes as long a minute. The training and testing speech segments were collected using different handsets. In some instances these handsets were of the same type (both electrets or both carbon) and sometimes they differed [18, 100, 101].

6.2.1 The NIST 2000 One-Speaker Detection Task

The one speaker detection task in the NIST 2000 speaker recognition evaluation is set up as follows. Each test speech segments is allocated eleven speaker models against which it is to be tested. Of these speaker models, ten are non-target/impostor trials and one correctly matches the test segment. The evaluation kit provides an answer key from which one can determine which of the speaker models is the target model.

In this work only a small subset of the trials included in the NIST 2000 evaluation was performed. Due to the long training time associated with the support vector machine, twenty trial sets were used for the experimental work. There were therefore twenty target trials and 200 non-target trials performed, resulting in a total of 220 trials in each experiment. The trials carried out involved only female speakers.

6.3 Software Toolkits

This section discusses the software tools which were used in the experiments which were carried out in this research.

6.3.1 Edinburgh Speech Tools

The Edinburgh Speech Tools (EST) [102] toolbox is a collection of C++ classes, functions and related programs designed and made freely available by the Centre of Speech Technology Research at the University of Edinburgh. The toolbox includes a selection of speech processing classes, some of which are the underlying classes for the Festival Speech Synthesis system. This toolbox was used for generating the speech features for this research as well as the pre-processing such as framing, windowing and the pre-emphasis filter stage.

6.3.2 SVMTorch

The SVMTorch [103, 104] engine created by Institut Dalle Molle d'Intelligence Artificielle Perceptive (IDIAP) Research Institute, was used for the classification engine. Other popular SVM toolkits are LibSVM [105] and SVMLight [106]. A comparison of the performances of the different SVM engines was beyond the scope of this research.

The gaussian radial basis kernel function shown in Table 3.1 was used with $\sigma = 1$ and the SVM error-cost parameter was set to $C = 256$. These values are drawn from [13] where it was found that these parameter values resulted in good performance on a speaker identification task.

Finding the optimum values for the SVM parameters was beyond the scope of the research presented in this thesis as the main aim of the project was to establish the effect of the various techniques mentioned in preceding chapters on the performance of the SVM. The optimisation of these parameters would have been a time-intensive exercise. It is an intuitive assumption that any improvements or degradations to the performance that are experienced by a non-optimal system used here would be experienced similarly in a system with optimised SVM and kernel parameters.

6.3.3 FastICA

FastICA [107] is a freely available software package that implements Hyvärinen's fast fixed-point algorithm for independent component analysis. It is primarily a MATLAB package but versions which support C++ and Octave are also available. The MATLAB based version 2.5, published in October 2005, was used for this project. The software was developed at the Laboratory of Computer and Information Science (CIS) in the Department of Computer and Science Engineering, Helsinki University.

6.4 Background Speaker Models

Two background models were created using the NIST 1999 SRE database. A different database was used for the experiment database in order to avoid introducing a bias to the system. The background models were created to be gender and handset dependent:

1. FEMALE/ELEC
2. FEMALE/CARB

Each speaker's model was then trained with the appropriate gender/handset model to match their training data. Literature generally agrees that using handset and gender dependent background models improves the system performance [29, 38] .

Each background model consisted of roughly an hour of speech from a random selection of speakers. The specific information relating to the background models is shown in the table below.

Table 6.1: Background speaker model specifications

Gender	Handset Type	Length [seconds]	Number of Speakers
Female	Carbon	3624.91	111
Female	Electret	3643.35	125

6.5 Measuring System Performance

The performance of the speaker verification system was measured using the Detection Error Trade-off (DET) [43] curve and the Equal Error Rate (EER). The DET curve plot the False Acceptance Rate (FAR) against the False Rejection Rate (FRR) on a logarithmic scale while the EER is the point where the FAR and the FRR values are equal. A lower EER value typically corresponds to a better system performance. The

DET curves in this thesis were generated using MATLAB-based tools available from [108].

6.6 System Configuration

The components of the speaker verification system used in this thesis were configured as shown in figure 6.1.

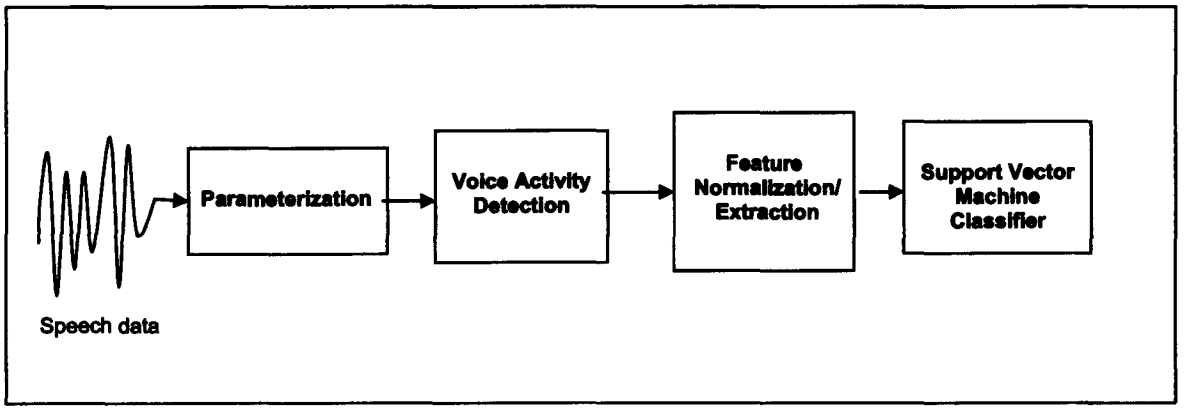

Figure 6.1: Speaker verification system component configuration

Following the general pre-processing procedure described in Chapter Two a pre-emphasis filter with $\alpha = 0.97$ was used. In framing, the frame length used was 25ms with a 10ms overlap and windowing was performed using a Hamming window. Thereafter energy-based voice activity detection was performed in order to remove the portions of the segment which carried silence. In [109] the authors recommend scaling the features before applying SVM. This avoids numerical problems which might be caused by large attributes in the feature vector. The recommended scaling range is [-1, +1] or [0,1]. In this work the feature attributes were scaled by

$$x' = \frac{x}{\max(abs(\mathbf{x}))} \tag{6.1}$$

where x' is the scaled attribute.

In [13] this scaling of the feature attributes was shown to improve the performance of an SVM classifier used a speaker identification task. Scaling was performed on the baseline system as well as in the evaluation of feature normalization. In evaluating feature extraction scaling was not applied because, as mentioned previously, the extracted features are not invariant under transformation and even scaling the attributes can change the resulting features.

Feature normalization and feature extraction were then performed where required, depending on the experiment in question. The output of this process was then used to train the support vector machine classification engine.

6.7 Chapter Summary

This chapter provided a discussion of the speech corpus and software tools used in this research. It also gave a review of the experimental procedures and system configuration used in this work. The results and analysis thereof are given in the next chapter.

CHAPTER SEVEN

7. RESULTS

This chapter presents the results obtained in the experiments described in the preceding chapters.

7.1 Selecting a Feature Set

In order to select the feature set to use for the experiments presented in this thesis the performance of LPCC and MFCC features were compared. Two speaker verification tasks were performed. In the first, the speakers training and testing handset types were matched (electret/electret) and in the second task the handset types were mismatched (electret/carbon). For the remainder of this thesis "mismatched handset" will refer to instance where the handset types used in training and testing differ and "matched handset" refers to instances where the handset types used in training and testing phases are the same. As mentioned in the previous chapter, the NIST 2000 corpus is configured such that while the handset type sometimes differed and sometimes did not, all training and testing speech segment were collected using different physical handsets.

Each of ten test speech segments was used in ten impostor trials and one target trial. No noise compensation or feature extraction was performed for these initial trials as the goal was simply to establish which feature set would be most appropriate for the remainder of the thesis work.

Each feature set was tested using a 17-dimensional feature vector. Pre-emphasis filtering was applied in both cases with the filter coefficient value set to 0.97. A filterbank order of 26 was used for the MFCC feature set while a 14-order Linear Predictive Analysis was carried out for the LPCC feature set. A simple energy-based voice activity detection scheme based was applied which removed 30-40% of the speech frames. Delta and delta-delta features were not used in either case.

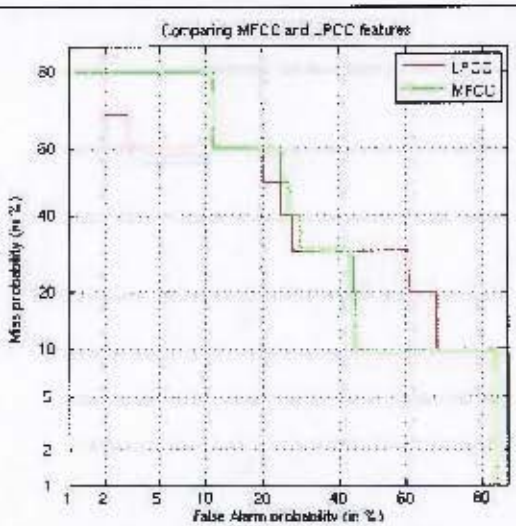

Figure 7.1a: Comparing LPCC and MFCC feature set performance on matching training and testing handsets.

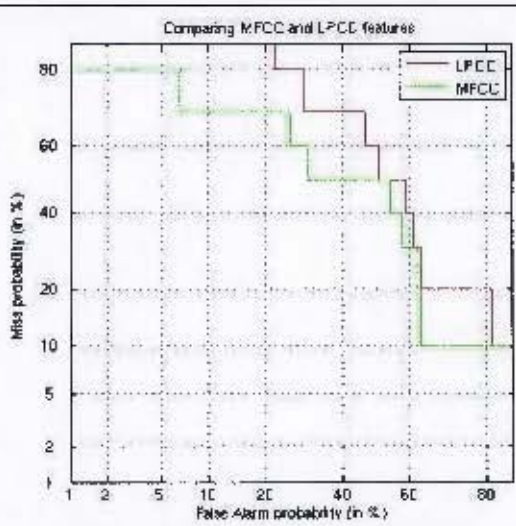

Figure 7.1b: Comparing LPCC and MFCC feature set performance on mismatched training and testing handsets.

Figure 7.1: Comparing the performance of LPCC and MFCC features.

On comparing the performances of the LPCC and MFCC features in a baseline system it was found that handset mismatches had an adverse effect on the performances of both

feature sets. However, since the MFCC feature set is the more dominant in terms of its representation in literature and research it was the feature set used in this research.

7.2 The Effect of Feature Normalization

One of the objectives of this research is to compare the effect that different feature normalization techniques have on system performance. To this end, the well established Cepstral Mean Subtraction and Mean Variance Normalization techniques were compared to Histogram Normalization which has only recently been applied to speaker verification [18]. The following sections discuss the verification of the HEQ algorithm used and then present the results achieved with the feature normalization techniques. The CMN and MVN algorithms are much simpler than HEQ to implement. CMN simply requires the subtraction of the long-term mean from each feature vector while MVN expands on this principle by normalizing the variance to unity. Consequently, an in-depth discussion of the verification of these algorithms is not included.

7.2.1 Verifying the HEQ Algorithm

In order to verify that the HEQ algorithm was implemented correctly it was tested on the popular Timit and NTimit databases [110, 111] following similar procedures as those used in [18].Timit is an *acoustic-phonetic continuous speech corpus* [110]. The speech data in this corpus is clean while NTimit consists of the same speech data as Timit but having been collected over the telephone network. NTimit is thus a contaminated version of Timit.

As mentioned in Section 4.4, the basic idea behind HEQ is to match the distribution of the training and testing data in order to compensate for any distortion that might have been introduced due to mismatched training and testing conditions.

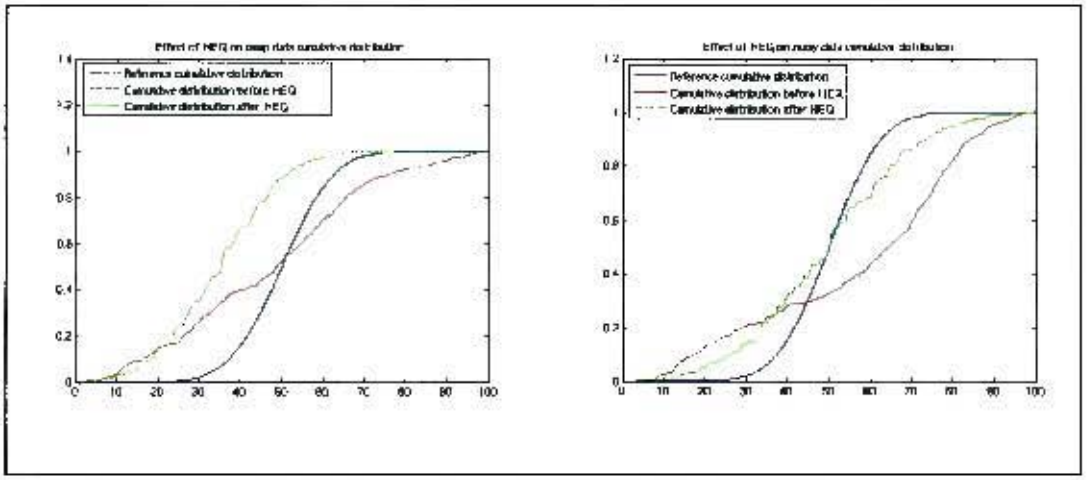

Figure 7.2: Effect of HEQ on cumulative distributions for clean and noisy speech data.

Figure 7.2 shows the cumulative distribution of the first MFCC feature vector for the clean and noisy version of a particular utterance taken from the Timit and NTimit databases, respectively. It is clear that the histogram normalization process brings both the clean and noisy data distribution closer to the reference distribution and thus closer to each other. The reference distribution is a Gaussian (normal) distribution with zero mean and unit variance. Figures 7.3 and 7.4 echo this result. These figures show the histograms of the clean and noisy speech data feature vector, before and after applying HEQ, respectively.

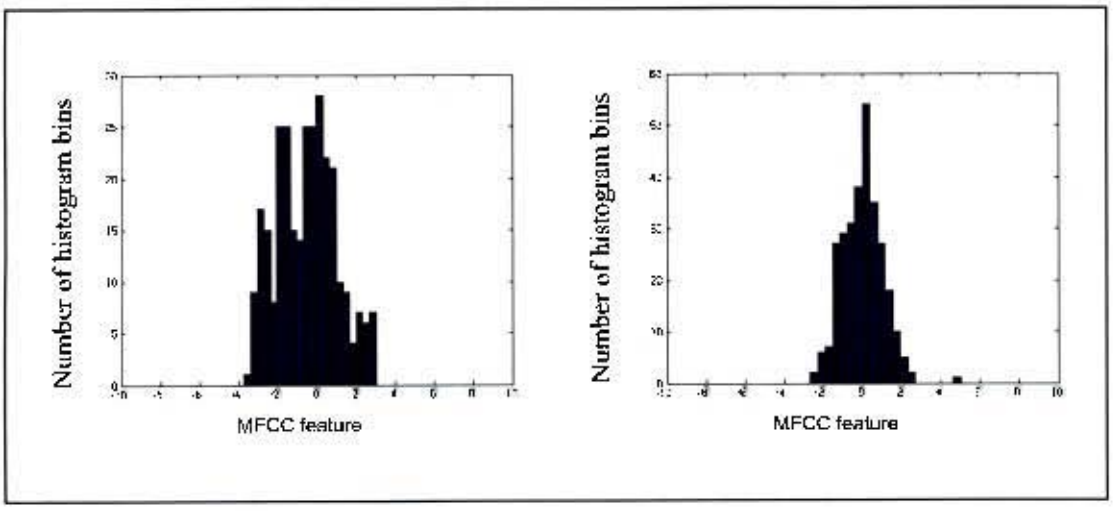

Figure 7.3: Effect of HEQ on MFCC feature vector histogram for clean data.

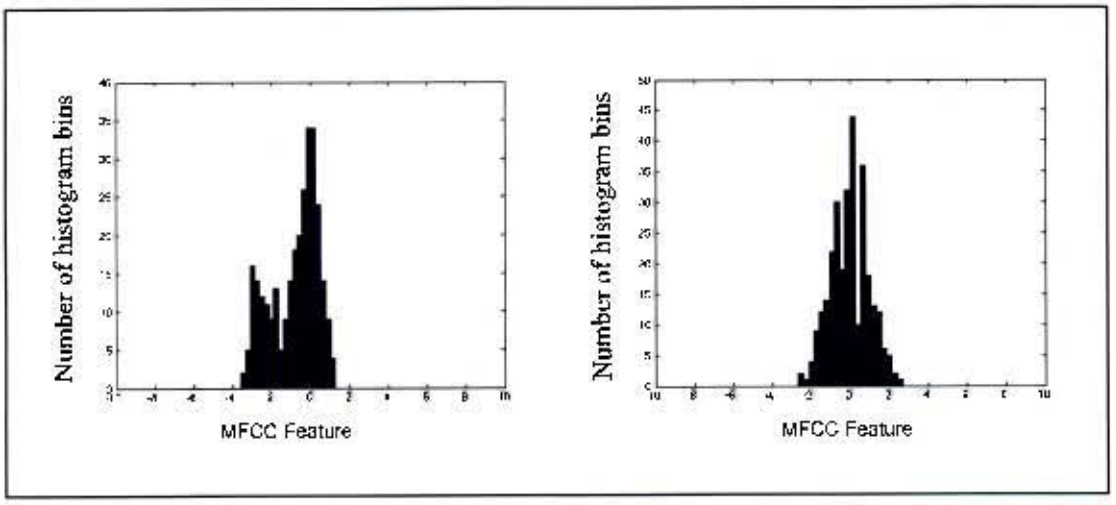

Figure 7.4: Effect of HEQ on MFCC feature vector histogram for noisy data.

As figures 7.3 and 7.4 show, the resultant histograms for both the clean and noisy data are more like each other in terms of shape and spread as well as being more like the reference normal distribution.

Plots of the trajectory of the MFCC feature vector before and after the application of HEQ are shown in figures 7.5 and 7.6. The figures show the trajectory of the vector extracted from clean and noisy speech data, respectively. As can be seen, HEQ has the

effect of improving the consistency of the feature vector distribution even in the presence of mismatch in the recording conditions.

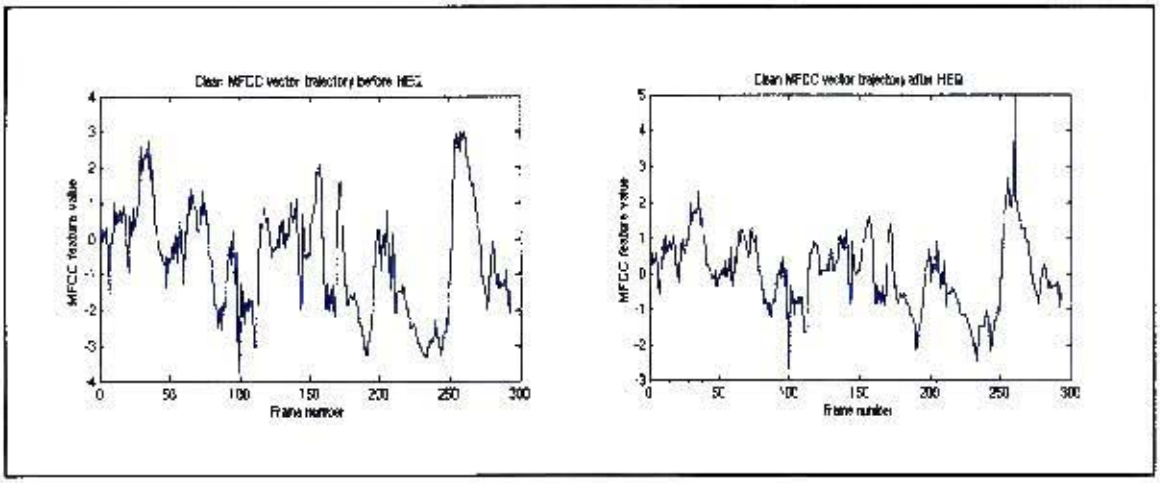

Figure 7.5: Effect of HEQ on trajectory of MFCC feature vector for clean data.

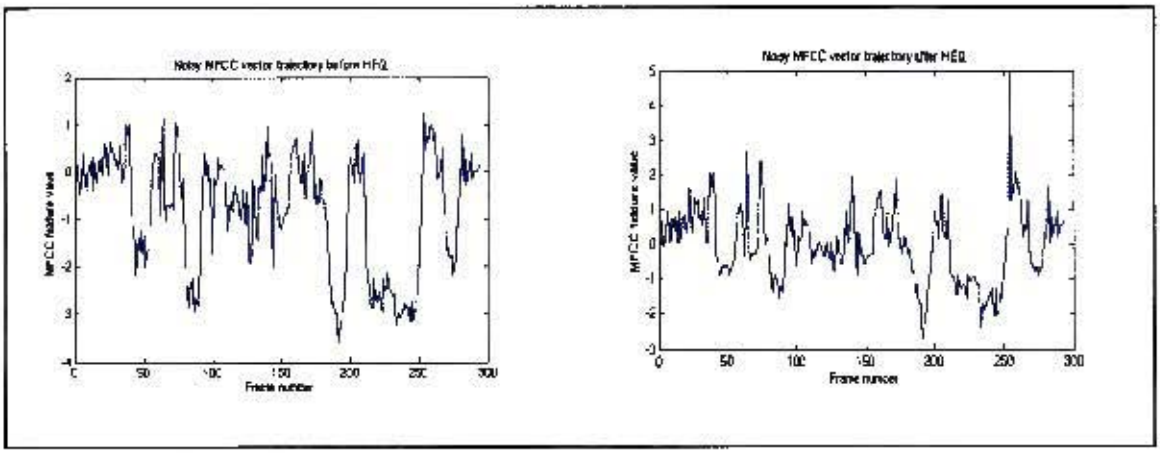

Figure 7.6: Effect of HEQ on trajectory of MFCC feature vector for noisy data.

7.2.2 Comparing Normalization Technique Performance

As explained in Section 4.1, Cepstral Mean Normalization requires the subtraction of the long term mean from the cepstral feature vectors resulting in vectors with zero mean while Mean Variance Normalization goes a step further by normalizing the vector

variances to unity. As in [72], the histograms used in the HEQ algorithm were estimated using 250 equally spaced bins.

The performance of the MFCC baseline system was compared to the system performance achieved by adding feature-based compensation algorithms. The normalization algorithms that were tested were Cepstral Mean Subtraction, Mean Variance Normalization and Histogram Normalization. The resultant DET curves using matched and mismatched training and testing handsets are shown in Figure 7.7 and 7.8, respectively. The EER achieved by each system is represented by the intersection of the corresponding DET curve with the dashed line.

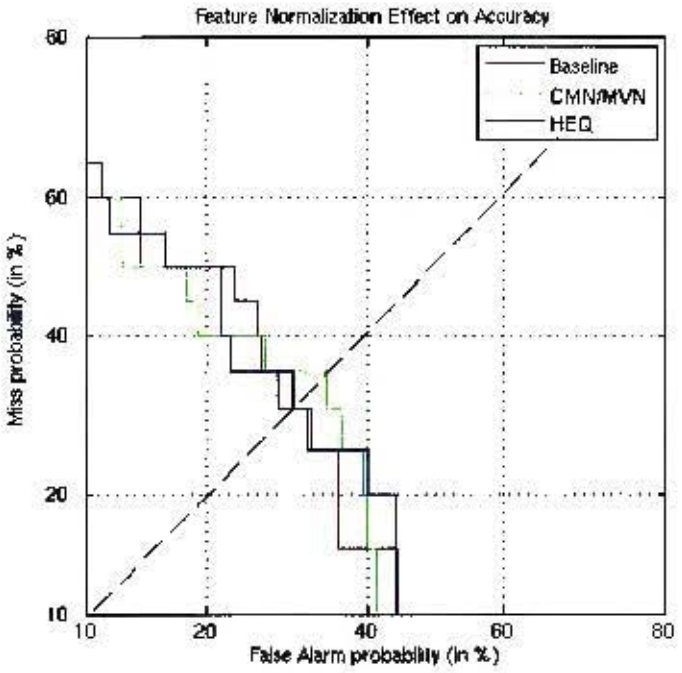

Figure 7.7: Resultant DET curve for matched training and testing handsets.

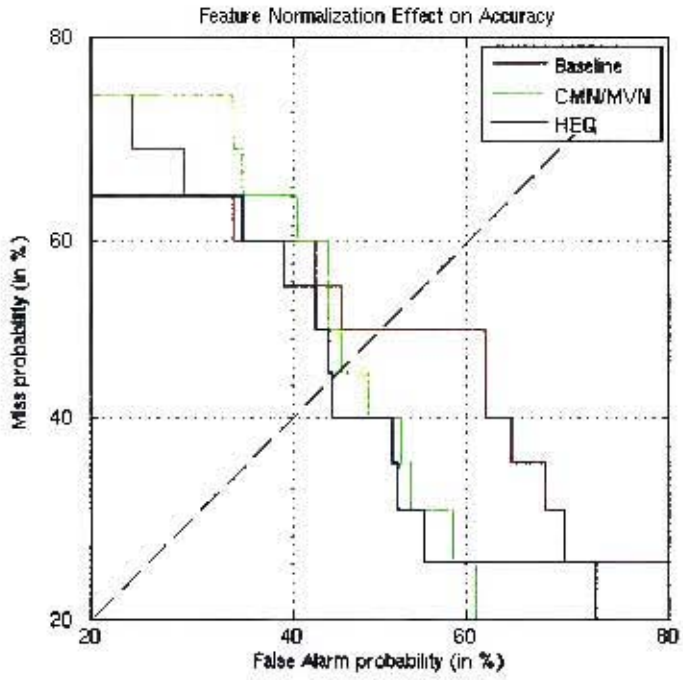

Figure 7.8: Resultant DET curve for mismatched training and testing handsets.

The results depicted by Figure 7.7 and 7.8 are echoed the Table 7.1. The table also contains the average size of the models generated by the SVM classifier during the training process as well as the average number of support vectors used in the models.

Table 7.1: Effect of feature normalization on equal error rates

Feature Normalization Type	**Matched Handset EER [%]**	**Mismatched Handset EER [%]**	**Average Size of Model [KB]**	**Average No. Support Vectors in Model**
Baseline	30.00	50.00	1945.60	10549.05
Cepstral Mean Normalization	34.50	45.50	419.80	3309.15
Mean Variance Normalization	34.50	45.50	419.80	3309.15
Histogram Equalization	30.00	44.50	320.64	2026.45

As the results show, applying feature normalization improves the accuracy of the SVM speaker verification system in the case of mismatched training and testing handsets, with HEQ resulting in the lowest (therefore best) equal error rate.

In the case of matched training and testing handsets, the application of CMN and MVN resulted in a degradation of the system accuracy while HEQ matches the baseline system performance. The similarity of the CMN and MVN results can be attributed to the scaling of the feature attributes which was performed in these experiments as discussed in section 6.6. Scaling the attributes effectively normalizes the variance of the feature vectors and since both CMN and MVN normalise the long-term mean this results in similar feature vectors and thus similar results. The application of HEQ also results in a reduction of the computer resources required for the storage of the speaker models.

7.3 Feature Extraction and Dimensionality Reduction

This section discusses the results achieved using feature extraction. First, PCA and ICA were individually applied to the feature extraction without reducing the data dimensionality. The results achieved were compared to the results achieved by the baseline system. The purpose of this was to determine which of these two feature transformation techniques would be used for the dimensionality reduction portion of the experiments to be performed. The effects of ICA and PCA on the SVM training time and system accuracy, with the data dimensionality remaining at the original 17, are shown in Table 7.2.

As mentioned in Section 5.2 there are a number of contrast functions and corresponding nonlinearities that can be used in the FastICA algorithm when performing ICA. Thus, in the case of ICA, three separate experiments were performed, one for each of the three nonlinearity functions discussed in Chapter Five. In order to ensure accurate comparison, computers with matching specifications were used for the experiments discussed in this

section. The computer models which were used were Intel Pentium 4 with a 3.20GHz central processing unit.

Table 7.2: Effect of PCA and ICA feature extraction on SVM performance

Feature Extraction Technique	Average Training Time [seconds]	Average Testing Time [seconds]	Matched Handset EER [%]	Mismatched Handset EER [%]	Average Size of Model [KB]	Average No. Support Vectors in Model
Baseline	23782.93±12912.05	3.97 ± 2.02	30.00	50.00	1945.60	10549.05
PCA	6401.59 ± 522.16	26.40 ± 11.89	90.00	95.00	12060.30	86445.05
ICA [pow3: $g(u)=u^3$]	12128.00 ± 138.38	63.24 ± 19.52	100.00	100.00	22738.05	162400.05
ICA [tanh: $g=\tanh(u)$]	12138.67 ± 172.88	64.52 ± 24.79	100.00	100.00	22738.05	161322.80
ICA [gauss: $g(u)=u\exp\left(\frac{1}{2}u^2\right)$]	12271.11 ± 416.11	66.53 ± 35.72	100.00	100.00	22732.80	161269.70

As can be seen in Table 7.2 the application of feature extraction immediately leads to a reduction of the average training time for the SVM models with PCA achieving the least average training time. It must also be that noted that the ICA algorithm often had trouble reaching convergence. This is in line which the findings reported in [94]. Some of the time savings achieved by the application of ICA would therefore be lost to processing time required for the convergence of the algorithm. Finally, the application of ICA led to more significant loss of accuracy, resulting in an EER of 100% in the case of mismatched training and testing handsets. This EER value means applying ICA led to every test segment being incorrectly classified. ICA also required double the computer resources that PCA required for the storage of the speaker models. From these results it was concluded that PCA feature extraction would be used to investigate the effect of dimensionality reduction on the SVM training time and accuracy.

7.3.1 Effect of Dimensionality Reduction on SVM Performance

From the results shown in the above section it was decided that PCA would be the technique used to investigate the effect the dimensionality reduction on the performance of the SVM. As noted in Section 5.1.1, PCA extracted features are not invariant under transformation and are sensitive even to scaling. As such, the PCA features were not scaled before processing with SVM. The results in this section show the effect on system accuracy and SVM training time as the data dimensionality is decreased. The results achieved with the application of feature extraction are compared to the results which were achieved with the baseline system as well as the results achieved with the application of HEQ. Note that neither the baseline nor the HEQ system had any form of feature extraction or dimensionality reduction applied.

Time constraints did not allow for every discrete value between the maximum dimension, 17, and the minimum, 1, to be used and so dimensions 2, 5, 8, 11, 14 and 17 were considered. These values were selected randomly in an attempt to give a clear indication of the effect of decreasing the data dimensionality in the range [1; 17].

The average training time for the baseline system was found to be 23782.93±12912.05 seconds while the average training time for the speaker verification system using HEQ feature normalization, which was the best performer in EER terms, was found to be 2209.28 ± 3642.04 seconds. Applying HEQ thus led to a 90% reduction in the average SVM training time. HEQ also led to a 47% decrease in the SVM testing time when compared to the baseline testing time. The time was measured using the Linux/Unix *time* command.

Table 7.3: PCA dimensionality reduction effect on SVM performance

Data Dimensionality	Average Training Time [seconds]	Average Testing Time [seconds]	Matched Handset EER [%]	Mismatched Handset EER [%]	Average Size of Model [KB]	Average No. Support Vectors in Model
Baseline (no PCA)	23782.93±12912.05	3.97 ± 2.02	30.00	50.00	1945.6	10549.05
HEQ (no PCA)	2209.28 ± 3642.04	2.12 ± 1.28	30.00	44.50	320.64	2026.45
2	19422.89 ± 5683.70	6.98 ± 5.71	50.00	56.50	696.32	29121.10
5	4799.20 ± 7174.88	2.03 ± 1.21	30.00	50.00	384.00	7710.80
8	1809.35 ± 167.45	9.79 ± 3.47	50.00	75.00	2035.12	30318.75
11	3624.19 ± 355.32	8.97 ± 6.28	85.00	95.00	5242.88	54507.25
14	5231.98 ± 453.21	23.32 ± 10.14	90.00	95.00	8770.06	73436.85
17	6401.59 ± 522.16	26.40 ± 11.89	90.00	95.00	12060.30	86445.05

As can be seen in Table 7.3, while reducing the dimension to 8 achieved the lowest average training time, the testing time was more that twice the average testing time of the baseline and equal error rate achieved was also worse than the baseline performance. When the data dimensionality was reduced to 5, the performance matched the results achieved by the baseline system while significantly reducing the average training time of the SVM classifier as well as reducing the average size of the speaker models. This result is in line with Wang's finding in [20] that the data dimensionality could be reduced to 60% – 70% of the original dimensionality without significant loss of information. In this task we were able to reduce the data dimensionality to 30% of the original dimensionality without degradation to system performance.

A drawback of the approach taken in this research is that there is no way to predetermine what the ideal dimensionality of data to be used is.

7.4 Chapter Summary

This chapter presented the results which were achieved during the work discussed in this thesis.

The results showed that applying HEQ to the feature vectors prior to training the SVM models improved the performance of the speaker verification system not only in terms of the accuracy but also by significantly decrease the average training time of the speaker models. Applying PCA feature extraction and reducing the data dimensionality from 17 to 5 decreased the training time and was able to match the performance of the baseline system. The reduced dimension performance was, however, not able to outdo the performance achieved by applying HEQ to the data. Applying HEQ to the features prior to SVM led to a 90% decrease in average training time required and a 47% decrease in average testing time. The results show that SVM classification is sensitive to the data pre-processing not only in terms of the accuracy but also the computation time that is required both in training and testing.

The results also seem to suggest that there is a relationship between the number of support vectors in the SVM model and the accuracy of classification. An intuitive interpretation of this relationship seems to be that the error in classification decreases as the number of support vectors decrease. However, Burges [19] points out that the intuitive conclusion that fewer support vectors give better performance is very often not true. So, while it might seem to hold true for the work conducted in this research, it ma very well fail when applied to other tasks.

CHAPTER EIGHT

8. CONCLUSIONS AND RECOMMENDATIONS

This chapter presents a summary of the work carried out in this study as well as the results which were achieved. It also consists of a discussion of future avenues of research related to this research.

8.1 Summary of Work

The experimental work carried out in this thesis was categorised into two main sections. The first focussed on the effect of feature normalization on the performance of an SVM based speaker verification system. A baseline system was compared to systems which used CMN, MVN and HEQ feature normalization.

The second aim of the project was to determine the effect of feature extraction and dimensionality reduction on the SVM training time and speaker verification accuracy.

Two linear feature extraction techniques, PCA and ICA were compared. The ICA algorithm used was the FastICA algorithm developed by Hyvärinen.

8.2 Summary of Results

The results reported in the previous chapter can be summarised as follows:

- The application of HEQ feature normalization significantly improved the EER as compared to the baseline in the case of mismatched training and testing handsets. Both CMN and MVN also improved the system performance when compared to the baseline but the HEQ algorithm outperformed both linear techniques. There was no notable difference in the system performance achieved with CMN and MVN.
- When matching handset types were used for training and testing HEQ was able to achieve the same EER as the baseline while both CMN and MVN resulted in a worsened system performance.
- The application of HEQ reduced the resources required to store the speaker models. With HEQ applied, the average memory space required for the speaker models was less than twenty percent of the space required by the baseline.
- Applying HEQ also reduced the average time taken to train speaker models. The average training time for speaker models with HEQ applied was approximately a tenth of the average training time achieved by the baseline system.
- Although applying ICA feature extraction led to a decrease in the average SVM training time (as compared to the baseline), it led to a significant increase in testing time and a loss of accuracy.
- Applying PCA feature extraction achieved a lower average SVM training time then both the baseline and the application of ICA. However, the transformation did lead to degradation in the system performance until the data dimensionality was reduced to five. At this point the baseline system was matched with

significant reductions in the SVM training time. This approach was however not able to match the performance achieved by applying HEQ feature normalization.

8.3 Recommendations for Future Research

This section discusses some possible avenues for future research which might improve on the results presented in this thesis.

8.3.1 Determining Optimum SVM Parameters

Optimizing the parameters of the SVM kernel would allow for better comparison of SVM with current state-of-the-art algorithms. This is important for the development of SVM and related research fields as it would enable a precise measure of the benefits and/or shortfalls of SVM classification. However it must be noted that research has shown that the SVM parameter end to be data-dependant, i.e. there is no parameter set that will give optimum results on every dataset.

8.3.2 Ordering the ICA Components

As mentioned in Chapter Five, ICA does not lend itself as easily as PCA to the ordering of the components. While in PCA the eigenvalues can be used to impose an order on the corresponding eigenvectors, this is not as simple in ICA. In [94] a class separability criterion was used to order the vectors. The criterion used, $r_i = \sigma_{bi} / \sigma_{wi}$, is similar to the class separability criterion used in LDA with σ_{bi} and σ_{wi} being the between and within class variance of the i-th component respectively. Imposing this type of ordering on the ICA components might be a way to improve the ICA components and thus improve the performance of ICA in this task.

8.3.3 Supervised Feature Extraction

In [112] an ICA-based feature extraction algorithm which includes the output class information is proposed. The advantage of this technique as compared to original ICA is that the class information is now taken into account during the ICA transformation. The additional information compensates for the unsupervised nature of ICA. The class information is included by treating the class label as an additional feature and adding it to the data before performing ICA. Thus data which originally consisted of N features would now have $N+1$ features. The algorithm was applied to data provided in the University of California (UCI) repository of machine learning datasets. The results showed an overall improvement in classification performance when the proposed algorithm was applied as opposed to using the original features without the feature extraction.

Similarly, the work in [113, 114] applied a supervised method of ICA to handwritten digit recognition. The method is based on Supervised ICA (SICA) as proposed in [115]. In this supervised version of ICA a separation matrix is trained such that the contributions of the independent components can be controlled by a supervisor. The training of SICA is carried out by maximising the correlations between each independent component and specific sets of inputs as well as by maximising the independence of the independent components. The technique was applied to a hand-written digit recognition task and it was found that the introduction of supervision to the feature extraction was effective and *within-class* to *between-class* variance ratio increased.

These methods can clearly be extended to speaker recognition tasks as well. It is a worthwhile research topic to determine whether the inclusion of class information improves the performance of ICA feature extraction in speaker verification. A comparison of the different approaches to including the class information would also be meaningful.

8.3.4 Nonlinear Feature Extraction

As mentioned, ICA and PCA are linear feature extraction techniques. Kernel ICA and Kernel PCA are nonlinear versions of the linear algorithms, which make use of the kernel trick described in Section 3.4. These are able to represent nonlinear relationships within the data.

KPCA speech feature extraction maps the conventional speech feature vectors to a feature space via a nonlinear mapping, performed implicitly using the kernel trick, onto the principal components [79]. Some research exists that addresses the application of KPCA in speech recognition tasks [79, 116]. The work presented in [116] reported that while applying PCA to a phoneme recognition task using an SVM classifier improved the system performance, the application of KPCA resulted in even higher accuracy. Applied to a handwritten digit recognition problem in [117] using a separating hyperplane classifier, KPCA was found to out perform the performance of its linear counterpart. However, neither of these studies considered the reduction of the data dimensionality. Research into the applicability of KPCA to the speaker verification task as described in this thesis would provide insight which might lead to a decrease into the SVM training time while lessening the loss of accuracy. This is true for the application of KICA as well.

8.4 Chapter Summary

This chapter discussed avenue of future research related to the work presented in this study. The results presented in the previous chapter show that the support vector machine is sensitive to the data representation and that the data pre-processing conducted affects not only the accuracy of SVM classification but also the speed of computation. The SVM has already been shown to have desirable characteristics such as good generalization and good classification performance. Reducing the training time associated with the SVM

will go a long way in allowing the technique to compete with present state-of-the-art approaches to speaker verification.

BIBLIOGRAPHY

[1] R. A. Cole, J. Mariani, H. Uszkoreit, A. Zaene, and V. Zue, "Survey of the State of the Art in Human Language Technology," Center for Spoken Language Understanding, Carnegie Mellon University, 1995.

[2] J. P. Campbell, "Speaker Recognition: A Tutorial," *Proceedings of the IEEE*, vol. 85, pp. 1437-1462, 1997.

[3] T. Kinnunen, "Spectral Features for Automatic Text-Independent Speaker Recognition," Department of Computer Science, University of Joensuu, Finland, 2003.

[4] A. Schmidt-Nielsen and T. H. Crystal, "Speaker Verification by Human Listeners: Experiments Comparing Human and Machine Performance Using the NIST 1998 Speaker Evaluation Data," *Digital Signal Processing*, vol. 10, pp. 249-266, 2000.

[5] H. Ezzaidi and J. Rouat, "Speaker Identification by Computer and Human Evaluated on the SPIDRE Corpus," *Canadian Acoustics*, vol. 28, pp. 156-157, 2000.

[6] A. K. Jain, A. Ross, and S. Prabhakar, "An Introduction to Biometric Recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, pp. 4-20, 2004.

[7] D. A. Reynolds, "Automatic Speaker Recognition: Current Approaches and Future Trends," presented at Workshop on Automatic Identification Advanced Technologies, New York, USA, 2002.

[8] T. W. Parsons, *Voice and Speech Processing*: McGraw-Hill, 1986.

[9] V. Vapnik, *The Nature of Statistical Learning*, Second Edition: Springer, 1999.

[10] V. Wan, "Speaker Verification Using Support Vector Machines," Department of Computer Science, University of Sheffield, 2003.

[11] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-Carrasquillo, "Support Vector Machines for Speaker and Language Recognition," *Computer Speech and Language*, vol. 20, pp. 210-229, 2005.

[12] A. Ganapathiraju, "Support Vector Machines for Speech Recognition," Department of Electrical and Computer Engineering, Mississippi State University, 2002.

[13] R. Jhumka, "Evaluation of Different Support Vector Machine Kernels," Department of Electrical Engineering, University of Cape Town, 2004.

[14] L. Quan and S. Bengio, "Hybrid Generative-Discriminative Models for Speech and Speaker Recognition," Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP) 2002.

[15] F. d. Wet, "Automatic Speech Recogniton in Adverse Acoustic Conditions," Department of Language and Speech, University of Nijmegen, 2003.

[16] R. J. Mammone, X. Zhang, and R. P. Ramachandran, "Robust Speaker Recognition-A Feature Based Approach," in *IEEE Signal Processing*, vol. 13, pp. 58-71, 1996.

[17] H. Hermansky and N. Morgan, "RASTA Processing of Speech," *IEEE Transactions on Speech and Audio Processing*, vol. 2, pp. 578-589, 1994.

[18] M. Skosan, "Histogram Equalization for Robust Text-Independent Speaker Verification in Telephone Environments," Department of Electrical Engineering, University of Cape Town, 2005.

[19] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.

[20] X. Wang, "Feature Extraction and Dimensionality Reduction in Pattern Recognition and Their Application in Speech Recognition," School of Microelectrical Engineering, Faculty of Engineering and Information Technology, Griffith University, 2002.

[21] J. P. Campbell, D. A. Douglas, and R. B. Duin, "Fusing High and Low-Level Features for Speaker Recognition," presented at European Conference on Speech and Communication Technology (Eurospeech), Geneva, Switzerland, 2003.

[22] C. Sanderson, "Automatic Person Verification Using Speech and Face Information," School of Microelectrical Engineering, Faculty of Engineering and Information Technology, Griffith University, 2002.

[23] "http://www.telecom.tuc.gr/~ntsourak/tutorial_acoustic_files/vclsys2.gif." last accessed June 2007.

[24] S. Kararekar, L. Ferrer, A. Venkataraman, K. Sonmez, E. Shriberg, A. Stolcke, H. Bratt, and R. R. Gadde, "Speaker Recognition Using Prosodic and Lexical Features," presented at IEEE Workshop on Automatic Speech Recognition and Understanding, St. Thoman, U.S. Virgin Islands, 2003.

[25] D. Reynolds, W. Andrews, J. Campbel, J. Navrati, B. Peskin, B. Adami, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, and B. Xing, "The SuperSID Project: Exploiting High-level Information for High-Accuracy Speaker Recognition," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2003.

[26] E. Shriberg, L. Ferrer, S. Kajerekar, S. Venkataraman, and A. Stolcke, "Modeling Prosodic Feature Sequences for Speaker Recognition," in *Speech Communication*, vol. 46, pp. 455-472, 2005.

[27] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*: Prentice Hall, 1993.

[28] J. Holmes and W. Holmes, *Speech Synthesis and Recognition*, Second Edition: Taylor and Francis, 2001.

[29] F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-Garcia, D. Petrovska-Delacretaz, and D. A.

Reynolds, "A Tutorial on Text-Independent Speaker Verification," *EURASIP Journal on Applied Signal Processing*, vol. 4, pp. 430-451, 2004.

[30] "http://www.owlnet.rice.edu/~elec532/PROJECTS98/speech/cepstrum/win1.jpg." last accessed June 2007.

[31] D. J. Mashao and M. Skosan, "Combining Classifier Decisions for Robust Speaker Identification," *Pattern Recognition*, vol. 39, pp. 147-155, 2006.

[32] C. Sanderson, "Speech Processing and Text-Independent Automatic Person Verification," Martigny, Switzerland, IDIAP Communication 02-08 December 2002.

[33] A. Ouzounov, "Robust Features for Speech Detection- A Comparative Study," presented at the International Conference on Computer Systems and Technologies (CompSysTech), Varna, Bulgaria, 2005.

[34] T. D. Ganchev, "Speaker Recognition," Wire Communications Laboratory, Department of Computer and Electrical Engineering, University of Patras, 2005.

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

[36] S. Umesh, L. Cohen, and D. Nelson, "Frequency Warping and the Mel Scale," *IEEE Signal Processing Letters*, vol. 9, pp. 104-107, 2001.

[37] N. T. Baloyi, "A Comparison of EIH and Mel-Cepstrum for Large-Vocabulary Text-Independent Speaker Identification Over Telephone Speech," Department of Electrical Engineering, University of Cape Town, 2000.

[38] D. A. Reynolds, "Speaker Identification and Verification Using Gaussian Mixture Models," in *Speech Communication*, vol. 17, pp. 91-108, 1995.

[39] H. Abdi, "A Neural Network Primer," *Journal of Biological Systems*, vol. 2, pp. 247-283, 1994.

[40] C. P. Lim, S. C. Woo, A. S. Loh, and R. Osman, "Speech Recognition Using Artificial Neural Networks," presented at International Conference on Web Information Systems Engineering, Hong Kong, China, 2000.

[41] R. V. Pawar, P. P. Kajave, and S. N. Mali, "Speaker Identification using Neural Networks," *Transactions on Engineering, Computing and Technology*, vol. 7, pp. 429-433, 2005.

[42] C. Cortes and V. Vapnik, "Support Vector Networks," *Machine Learning*, vol. 20, pp. 273-297, 1995.

[43] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET Curve in Assessment of Detection Task Performance," presented at European Conference on Speech Communication and Technology (Eurospeech), Rhodes, Greece, 1997.

[44] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*: Cambridge University Press, 2000.

[45] B. Schölkopf, C. J. C. Burges, and A. J. Smola, *Advances in Kernel Methods-Support Vector Learning*: MIT Press, 1999.

[46] A. Shilton, "Design and Training of Support Vector Machines," Department of Engineering, University of Melbourne, 2006.

[47] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," presented at Workshop on Computational Learning Theory, Pittsburgh,PA, 1992.

[48] L. Wiskott, "Lecture Notes on Support Vector Machines for Classification," http://itb1.biologie.hu-berlin.de/~wiskott/Teaching/SupportVectorMachines.pdf, 2006, last accessed June 2007.

[49] S. R. Gunn, "Support Vector Machines for Classification and Regression," University of Southampton, Technical Report May 1998.

[50] J. Salomon, S. King, and M. Osborne, "Framewise Phone Classification Using Support Vector Machines," presented at International Conference on Spoken Language Processing (ICSLP), Denver, Colorado, USA, 2002.

[51] P. Erasto, "Support Vector Machines-Backgrounds and Practice," Rolf Nebanlinna Institute, 2001.

[52] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing Multiple Parameters for Support Vector Machines," *Machine Learning*, vol. 46, pp. 131-159, 2002.

[53] K. Duan, S. S. Keerthi, and A. N. Poo, "Evaluation of Simple Performance Measures for Tuning SVM Hyperparameters," *Neurocomputing*, vol. 51, pp. 41-59, 2003.

[54] E. J. R. Justino, F. Bortolozzi, and R. Sabourin, "A Comparison of SVM and HMM Classifiers in the Off-line Signature Verification," *Pattern Recognition Letters*, vol. 26, pp. 1377-1385, 2005.

[55] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," University of Dortumund November 1997.

[56] B. Schölkopf, K.-K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2758-2765, 1997.

[57] M. S. Schmidt, "Identifying Speakers with Support Vector Machines," presented at Interface, Sydney, Australia, 1996.

[58] K. K. Chin, "Support Vector Machines Applied to Speech Pattern Classification," Computer Speech and Language Processing, Engineering Department, University of Cambridge, 1999.

[59] T. S. Jaakkola and D. Haussler, "Exploiting Generative Models in Discriminative Classifiers," *Advances in Neural Information Processing*, vol. 2, pp. 487-493, 1998.

[60] W. Campbell, "Generalised Linear Discriminant Sequence Kernels for Speaker Recognition," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), Orlando, Florida, USA, 2002.

[61] " NIST Speech Group," http://www.nist.gov/speech/index.htm last accessed June 2007.

[62] L. Quan and S. Bengio, "Client Dependent GMM-SVM Models for Speaker Verification," presented at International Conference on Artificial Neural Networks, Istanbul, Turkey, 2003.

[63] A. Ganapathiraju and J. Picone, "Hybrid SVM/HMM Architectures for Speech Recognition," *In Neural Information Processing Systems*, 2000.

[64] P. J. Moreno, "Speech Recognition in Telephone Environments," Department of Electrical and Computer Engineering, Carnegie Mellon University, 1992.

[65] J. P. Openshaw and J. S. Mason, "On the Limitations of Cepstral Features in Noise," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), Adelaide, Australia, 1994.

[66] S. v. Vuuren, "Comparison of Text-Independent Speaker Recognition Methods on Telephone Speech with Acoustic Mismatch," presented at International Conference on Spoken Language Processing (ICSLP), Philadelphia, USA, 1996.

[67] A. d. l. Torre, J. C. Segura, A. M. Peinado, A. J. Rubio, J. L. Perez-Corrdoba, and M. C. Benitez, "Histogram Equalization of Speech Representation for Robust Speech Recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 13, pp. 355-366, 2005.

[68] R. M. Stern, F.-H. Liu, Y. Ohshima, T. M. Sullivan, and A. Acero, "Multiple Approaches to Robust Speech Recognition," presented at Speech and Natural Language Workshop, Defense Advanced Research Projects Agency, 1992.

[69] F.-H. Liu, P. J. Moreno, R. M. Stern, and A. Acero, "Signal Processing for Robust Speech Recognition," in *Automatic Speech and Speaker Recognition: Advanced Topics*: Kluwer Publishing, pp. 357-384, 1996.

[70] Y. Obuchi and R. M. Stern, "Normalization of Time Derivative Parameters using Histogram Equalization," presented at European Conference on Speech Communication and Technology (Eurospeech), Geneva, Switzerland, 2003.

[71] J. C. Segura, C. Benitez, A. d. l. Torre, A. J. Rubio, and J. Ramirez, "Cepstral Domain Segmental Nonlinear Feature Transformations for Robust Speech Recognition," *IEEE Signal Processing Letters*, vol. 11, pp. 517-520, 2004.

[72] M. Skosan and D. Mashao, "Modified Segmental Histogram Equalization for Robust Speaker Verification," *Pattern Recognition Letters*, vol. 27, pp. 479-486, 2006.

[73] A. d. l. Torre, J. C. Segura, A. M. Peinado, and A. J. Rubio, "Non-linear Transformations of the Feature Space for Robust Speech Recognition," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), Orlando, Florida, USA, 2002.

[74] L. Garcia, J. C. Segura, J. Ramurez, A. d. l. Torre, and C. Benitez, "Parametric Nonlinear Feature Equalization for Robust Speech Recognition," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toulouse, France, 2006.

[75] S. Dharanipragada and M. Padmanabhan, "A Nonlinear Unsupervised Adaptation Technique for Speech Recognition," presented at International Conference on Spoken Language Processing, Beijing, China, 2000.

[76] A. Koscor, L. Tóth, A. Kuba, K. Kovács, M. Jelasity, T. Gyimóthy, and J. Csirik, "A Comparative Study of Several Feature Transformation and Learning Methods for Phoneme Classification," *International Journal of Speech Technology*, vol. 3, pp. 263-276, 2000.

[77] N. Kambhatla and T. K. Leen, "Dimension Reduction by Local Principal Component Analysis," *Neural Computation*, vol. 9, pp. 1493 - 1516, 1997.

[78] I. T. Jollife, *Principal Component Analysis*: Springer-Verlag, 1998.

[79] A. Lima, H. Zen, Y. Nankaku, C. Miyajima, and K. Tokuda, "On the Use of Kernel-PCA for Feature Extraction in Speech Recognition," *IEICE Transactions on Information and Systems*, vol. E87-D, pp. 2802-2811, 2004.

[80] J. Shlens, "A Tutorial on Principal Component Analysis," 2nd ed: http://www.snl.salk.edu/~shlens/pub/notes/pca.pdf., 2005, last accessed June 2007.

[81] L. I. Smith, "A Tutorial on Principal Component Analysis," http://csnet.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf, 2002, last accessed June 2007.

[82] A. Hyvarinen, "Survey on Independent Component Analysis," *Neural Computing*, vol. 2, pp. 94-128, 1999.

[83] N. Kwak and C.-H. Choi, "Feature Extraction Based on ICA for Binary Classification Problems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, pp. 1374 - 1388, 2003.

[84] J. Hurri, A. Hyvarinen, and E. Oja, "Wavelets and Natural Image Statistics," presented at Scandanavian Conference on Image Analysis (SCIA), Lappeenranta, Finland, 1997.

[85] J. V. Stone, *Independent Component Analysis: A Tutorial Introduction*: MIT Press, 2004.

[86] A. Hyvarinen and E. Oja., "Independent Component Analysis: Algorithms and Applications," *Neural Networks*, vol. 13, pp. 411-430, 2000.

[87] W. Zhen, L. Jin, and Y. Jin, "Independent Component Analysis: An Introduction." Department of Statistics, Stanford University.

[88] A. Hyvarinen, "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis," *IEEE Transactions on Neural Networks*, vol. 10, pp. 626-634, 1999.

[89] A. Hyvärinen and E. Oja, "Independent Component Analysis: Algorithms and Applications," *Neural Networks*, vol. 13, pp. 411-430, 2000.

[90] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*: John Wiley & Sons, 2001.

[91] H.-Y. Jung, M. Park, H.-R. Kim, and M. Hahn, "Speaker Adaptation Using ICA-Based Feature Transformation," *ETRI Journal*, vol. 24, pp. 469-472, 2002.

[92] G.-J. Jang, T.-W. Lee, and Y.-W. Oh, "Learning Statistically Efficient Features for Speaker Recognition," *Neurocomputing*, vol. 49, pp. 320-348, 2002.

[93] J.-H. Lee, H.-Y. Jung, T.-W. Lee, and S.-Y. Lee, "Speech Feature Extraction Using Independent Component Analysis," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), Istanbul, Turkey, 2000.

[94] C. Havran, L. Hupet, J. Czyz, J. Lee, L. Vandendorpe, and M. Verleysen, "Independent Component Analysis for Face Authentication," presented at Knowledge-Based Intelligent Information and Engineering Systems, Crema, Italy, 2002.

[95] G.-J. Jang, S.-J. Yun, and Y.-H. Oh, "Feature Vector Transformation Using Independent Component Analysis and its Application to Speaker Identification," presented at European Conference on Speech Communication and Technology, Budapest, Hungary, 1999.

[96] S. Gangisetty, "Text-Independent Speaker Recognition," College of Engineering and Mineral Resources, West Virginia University, 2005.

[97] L. J. Cao, K. S. Chua, W. K. Chong, H. P. Lee, and Q. M. Gu, "A Comparison of PCA, KPCA and ICA for Dimensionality Reduction in Support Vector Machine," *Neurocomputing*, vol. 55, pp. 321 - 336, 2003.

[98] G. Antonini, V. Popovici, and J.-P. Thiran, "Independent Component Analysis and Support Vector Machine for Face Feature Extraction," presented at International Conference on Audio and Video Based Biometric Person Authentication, Guiford, United Kingdom, 2003.

[99] Y. Qi, D. Doermann, and D. DeMenthon, "Hybrid Independent Component Analysis and Support Vector Machine Learning Scheme for Face Detection," presented at International Conference on Acoustics, Speech and Signal Processing (ICASSP), Salt Lake City, USA, 2001.

[100] J. Fiscus, W. M. Fisher, A. F. Martin, M. A. Przybocki, and D. S. Pallet, "2000 NIST Evaluation of Conversational Speech Recognition over the Telephone," presented at Speech Transcription Workshop, 2000.

[101] A. Martin and M. Przybocki, "The NIST Speaker Recognition Evaluations:1996 - 2001," presented at International Conference on Language Resources and Evaluation, Grenada, Spain, 1998.

[102] "The Edinburgh Speech Tools Library," http://www.cstr.ed.ac.uk/projects/speech_tools/, last accessed January 2007

[103] "SVMTorch," ftp.idiap.ch/pub/learning/SVMTorch.tgz, last accessed July 2006.

[104] R. Collobert and S. Bengio, "SVMTorch: Support Vector Machines for Large-Scale Regression Problems," *Journal of Machine Learning Research*, vol. 1, pp. 143-160, 2001.

[105] "LibSVM," http://www.csie.ntu.edu.tw/~cjlin/libsvm/ last accessed May 2007.

[106] "SVMLight," http://www.cs.cornell.edu/People/tj/svm_light/ last accessed May 2007.

[107] "FastICA," http://www.cis.hut.fi/projects/ica/fastica/ last accessed June 2007.

[108] "DETware_v2.1.tar.gz: DET-Curve Plotting Software," http://www.nist.gov/speech/tools/index.htm last accessed June 2007.

[109] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A Practical Guide to Support Vector Classification," http://www.csie.ntu.edu.tw/~cjlin/libsvm/ last accessed May 2007.

[110] "TIMIT: LCD Catalog," http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S1 last accessed May 2007.

[111] "NTIMIT: LCD Catalog," http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S2 last accessed May 2007.

[112] N. Kwak, C.-H. Choi, and J. Y. Choi, "Feature Extraction Using ICA," presented at International Conference on Artificial Neural Networks, Vienna, Austria, 2001.

[113] S. Ozawa, Y. Sakaguchi, and M. Kotani, "A Study of Feature Extraction Using Supervised Independent Component Analysis," presented at International Conference on Neural Networks (IJCNN), Washington DC, USA, 2001.

[114] S. Ozawa, Y. Sakaguchi, and M. Kotani, "Feature Extraction of Handwritten Characters Using Supervised and Unsupervised Independent Component Analysis," presented at World Multiconference on Systemics, Cybernetics and Informatics (WMSCI), Orlando, USA, 2001.

[115] S. Umeyama, S. Akaho, and Y. Sugase, "Supervised Independent Component Analysis and its Application to Face Image Analysis," Institute of Electronics, Information and Communication Engineers (IEICE), Technical Report NC99-2, 1999.

[116] A. Koscor, A. Kuba, and L. Tóth, "Phoneme Classification Using Kernel Principal Component Analysis," *Periodica Polytechnica - Electrical Engineering*, vol. 44, pp. 77-90, 2000.

[117] B. Schölkopf, A. Smola, and K.-R. Muller, "Kernel Principal Component Analysis," in *Advances in Kernel Methods - SV Learning*: MIT Press, Cambridge, MA, pp. 327-352, 1996.

[i] Source: [23] "http://www.telecom.tuc.gr/~ntsourak/tutorial_acoustic_files/vclsys2.gif."

[ii] Source: [30] "http://www.owlnet.rice.edu/~elec532/PROJECTS98/speech/cepstrum/win1.jpg."

[iii] Source: [87] W. Zhen, L. Jin, and Y. Jin, "Independent Component Analysis: An Introduction." Department of Statistics, Stanford University.