A Non-linear Polynomial Approximation Filter for Robust Speaker Verification

Prepared by:

Limpho Mothae

Supervised by:

Daniel J. Mashao

Deparment of Electrical Engineering

University of Cape Town

This dissertation is submitted to the University of Cape Town in fulfillment of the academic requirements for the degree of Master of Science in Engineering

9 December 2003

MOTHAE

Declaration

I declare that this project report is my own work. It is being submitted for the degree of Master of Science in Engineering at the University of Cape Town. It has not been submitted before for any degree or examination in any other university.

Signature of Author .....

Cape Town
9 December 2003

Acknowledgements

I am profoundly grateful to my supervisor, Dr. Daniel J. Mashao, for the assistance he provided throughout the execution of this project. I am especially grateful for the times that he helped me debug the code that I used for this project, and for his kindness in general. I am just as grateful to my parents for their relentless support and encouragement. My colleague and friend $\sum_{i=0}^{\infty} Lerato_i$ also deserves thanks for the laughs that we shared when the going was tough, and the help he always thought I needed.

I am just as grateful to Dr. Fred Nicolss for clarying certain signal processing techniques (when he was sober enough, that is). And to C. Bane for her love and support. Umm, sometimes. Last but not least, I am grateful to the NRF for their partial (sorry, financial) support. But above all I am eternally grateful to the One who watches over my soul. To Him belong honour and thanks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent advances in speech technology have enabled researchers to design speaker recognition architectures that are capable of sublime performance in ideal environments. Speaker recognition is a generic term for the classification of a speaker's identity using information extracted from a speech signal [20]. In contrast to speech recognition, whose main goal is to extract lexical information from a speech waveform, speaker recognition requires the extraction of unique parameters that individuate a speaker. It consists of speaker identification (SI) and speaker verification (SV). Both SI and SV can be text-dependent or text-independent. Text-dependent recognition requires the speaker to articulate phrases or sentences having the same text for both training and testing trials, whereas text-independent architectures have no such restrictions.

In addition, text-independent systems require more training data than text-dependent ones. This is necessary to ensure that a speaker's full range of vocal sounds are captured during training [45]. Typical text-independent recognition scenarios include a radio news broadcast and ordinary conversational speech. This form of recognition finds application in forensic investigations since investigators cannot force a suspect to utter a prescribed text. In addition to text-

dependent and text-independent methodologies, SV systems may also comprise challenge-response architectures [10]. In a challenge-response system, the user is prompted to articulate a randomly generated phrase in order to prevent criminals from recording or synthesizing a client's passphrase to obtain unauthorised access to their resources. Thus challenge-response SV architectures require lexical content to be interpreted prior to recognition, which necessitates that they comprise hybrid speech recognition / speaker verification architectures.

In the case of speaker identification, the speaker will be classified as being one of a finite set of enrolled speakers (closed-set) or external to that set (open-set). Therefore an SI system must make $1/n$ decisions, where n is the population size of registered speakers. Consequently, these systems' performance deteriorates as the speaker population grows. On the other hand, speaker verification entails the authentication or negation of an identity claim. Thus SV is more a discrimination process than an identification process [66]. Hence only two decisions have to be made at any given time. Needless to say, it is highly desirable for both SI and SV systems to consistently perform well irrespective of where the recognition task might be required.

Not surprisingly, the use of a person's voice to ascertain their identity has unparalleled appeal to the designers of biometric verification architectures. This is attributed to the fact that speech is the most natural means of communication for most people (as opposed to fingerprint recognition or retinal scanning, for instance). Furthermore, speaker recognition architectures can be designed and deployed relatively inexpensively. However, since speech is a very complex signal that conveys a considerable amount of information (e.g. approximate age, race, stature, gender, and information about the acoustic environment of the speaker), extracting time-invariant and environment-transparent parameters from it while suppressing unwanted signals which might include speech from other speakers (the so-called cocktail party effect [45]) is a non-trivial task.

1.1 Problem Statement

Although state-of-the-art speaker recognition architectures perform exceptionally well under ideal conditions [19], it is much harder for these systems to perform nearly as well in real-world scenarios. **However, this thesis will only focus on SV systems**. Some of the major causes of poor performance in real-world applications are detailed in the following subsections.

1.1.1 Mismatched Environments

Since a speech signal mainly conveys three different types of information [40] (i.e. linguistic, speaker-specific and environment information), it is hardly surprising that speaker verification architectures exhibit considerable sensitivity to the environments in which they operate. When a client speaks into a microphone, the resultant acoustic signal contains both speech and ambient noise, as well as delayed versions of the speech [21] generated by reverberant surfaces. These delayed versions of the speech signal are analogous to multipath fading in electromagnetic communication theory. Multipath fading arises when different versions of the same electromagnetic signal propagate along different paths towards the receiver, where they cause interference with the direct signal.

In an acoustic context, a sound wave will propagate from the speaker's mouth directly to the microphone, while other waves may first propagate to the wall(s) and be reflected back towards the microphone. As a result, the composite signal also conveys environmental information. Therefore when a speaker's model is created from a set of so-called training utterances, some information about the environment in which the training utterances were generated is incorporated into the model [60]. Consequently, when the speaker moves to another environment with different acoustic properties, successful verification can no longer be guaranteed. However, this does not imply that perfect performance may be expected

if the speaker remains in the same environment.

1.1.2 Handset Sensitivity

Handset variability occurs when training speech is collected using one type of handset, but a different handset is used for collecting speech [69]. In a far-field speaker verification scenario, the speech generated by a client is subjected to numerous distortions. Specifically, speech that has been transmitted across a telephony network is contaminated by ambient noise and convolved first with the transfer function of the telephone's microphone transducer and then with the transfer function of the telephone channel [64][69]. This explains the phenomenon of handset sensitivity. In other words, since different handsets have different transducers (and therefore different transducer impulse responses), it stands to reason that a speaker model that was created using a certain type of handset may not perform well when used to verify the speaker if a different handset is used, regardless of whether the acoustic environments corresponding to training (also known as enrollment) and testing (or recognition) are mismatched or not. The reason for this is that different transducer impulse responses will cause different convolutional distortions.

Many techniques such as cepstral mean subtraction (CMS), RASTA filtering and delta coefficients have been proposed and evaluated for handset sensitivity [69]. Nevertheless, the recognition performance for matched handset cases remains considerably higher than that for mismatched handsets despite the use of compensation algorithms [37].

1.1.3 Imposter Attacks

Unfortunately, SV systems are also vulnerable to deliberate attacks by malicious users, known as imposters. By definition, an imposter is a user who claims the

identity of another with the intention of gaining unauthorised access to their resources. As a matter of necessity, a good SV system must accept as many legitimate claims as possible while also rejecting as many imposter claims as possible. Owing to the fact that SV is a binary hypothesis test problem, it follows that two types of error can occur [21]. The first type of error is called the false rejection rate (FRR, also known as Type I error), while the second error type is called the false acceptance rate (FAR, otherwise known as Type II error). In practice these errors tend to be inversely related, meaning that one is reduced at the expense of the other.

Some SV methodologies assume that successful impostor attacks will originate only from similar-sounding clients, known as cohorts. This paradigm is clearly flawed. **This is because, in a fielded SV system, a user who is good at impersonating other speakers may speak naturally during enrolment but modulate their voice when targeting a specific victim during testing**. Thus the SV system would be vulnerable to imposters whose voices are very dissimilar to that of the victim since no consideration will have been given for them during the anti-speaker modelling phase. Thus one can expect that an SV system that was implemented based on the assumption that potential imposters sound like their victims might reflect optimistic results simply because no consideration was given for casual imposters who are external to the set of similar-sounding speakers. This suggests that it is difficult to estimate the real FAR of a fielded SV system in light of the fact that imposter attacks might originate from unexpected users.

Furthermore, it is likely that a determined imposter would target numerous victims and therefore modulate their voice differently with each attempt. Consequently, it may not be reasonable to extrapolate the performance of a fielded SV system based on simulations performed on an experimental database that does not comprise impersonation attempts. A comprehensive review of the most popular databases is given later in this chapter. Incidentally, none of these databases were

designed with notion of *adaptive impostors* in mind.

1.1.4 Speaker Variability

Voice has been proven to be a reliable indicator of a person's internal mood - emotion, health and personality, to mention but a few [75]. A limited database of 50 male Swedish speakers was created in [44], and two models were built for each speaker - one consisting of normal speech, and another consisting of emotional speech. It was found that the inclusion of emotional models improved performance by 32 %. Thus one can infer that people's voices tend to vary somewhat if any one of these psychological conditions also changes.

Unfortunately, it is quite difficult to develop an experimental database to study the effects of, say, ill-health or stress on a person's voice. For instance, a parent who has just lost a child (which is indicative of extreme stress) would not be interested in having samples of their voice recorded so that researchers could investigate the effects of stress on people's voices. Likewise, a person who is ill may feel irritable and therefore not want to cooperate in the development of such a database. In extreme cases a person may simply be too ill to speak.

On the other hand, a person who has just won a lottery may also be too elated to participate in the development of some "silly" database for obvious reasons. All these issues highlight the fact that the extent to which these psychological conditions affect a person's voice is difficult to determine objectively due to complications associated with collecting emotional or stressed speech. However, a handful of databases have been created [75], but one might argue that their usefulness is limited because of the impracticality of collecting sufficient quantities of such data. For instance, the Brno University[1] of Technology's Institute of Radio Electronics recorded a number of students during oral examinations, but it

[1]The university's url is http://www.vutbr.cz

is doubtful whether recording a few sessions per student would be enough to attribute the variability in a student's voice exclusively to stress. Furthermore, it is obvious that some students are more confident than others and thus may exhibit very subtle symptoms of stress, if any at all.

A typical corpus of stressed speech might be extracted from the cockpit voice recorder of a crashed plane [75], or a combat helicopter pilot under heavy anti-aircraft fire from hostile ground forces. However, it must be pointed out that the fidelity of data collected under these conditions is questionable since the pilot's speech would no doubt be masked by heavy background noise emanating from the aircraft's engine or missiles being fired from the aircraft, making it difficult to isolate the effect of stress. Moreover, such speech would also exhibit the Lombard effect. Thus it is still difficult to quantify the extent of distortion caused by stressed or emotional speech.

It is known that speaker recognition architectures perform poorly in real-world applications as a result of discrepancies between the assumed acoustic model of a speaker and the observed acoustic model [72]. According to this reference, these discrepancies are mainly caused by additive noise, channel distortion and an articulatory variability called the Lombard effect. The Lombard effect is explained simply by the fact that people tend to modify their voices in noisy environments so as to be heard above the noise [14]. Numerous model adaptation techniques have been proposed in an attempt to address these causes of variability individually. However, these said causes have a collective non-linear distortive effect on a speech signal [72]. Hence, using separate adaptation techniques to compensate for each factor individually does not solve the problem. This suggests that the only effective way to compensate for these distortions is to develop an adaptation model that addresses all the distortive elements (i.e. noise, channel distortion and the Lombard effect) simultaneously.

In the following section, a comprehensive review of the most common databases used in speaker recognition research is carried out.

1.2 State of the Art in SV

"Speaker verification has mainly found its way into niche applications such as access control" [11]. This is mainly due to the fact that the problems that were discussed in the previous section have not been completely solved. Consequently numerous corpora have been developed in an attempt to enable researchers to calibrate the performance of their architectures using databases that emulate real-life conditions. A comprehensive review of these databases is therefore carried out in the following subsections.

1.2.1 TIMIT and Derivatives

The TIMIT database was initially conceived as a platform for evaluating automatic speech recognition (ASR) systems. Owing to the fact that it has a lot of speakers (630 in total), it has also been used quite extensively for many speaker recognition studies. "The TIMIT family of corpora are useful for contrastive-type experiments to attempt to isolate and quantify the effect of specific degradations imposed on pristine data" [70].

Other renditions of TIMIT include: 1) FFMTIMIT (TIMIT recordings made from a far-field auxialiary microphone); 2) NTIMIT, which was created by articulating the sentences in TIMIT using an artificial mouth into a carbon-button telephone handset, transmitting the speech over short-haul and long-haul telephone lines and recording the received signal; 3) CTIMIT, generated by playing TIMIT speech into a cellular telephone handset in a moving vehicle, transmitting it over a mobile network and recording the signal at a remote destination; and 4) HTIMIT, which

was created by playing TIMIT speech through various telephone handset types and recording the signal directly form the output of the handset.

1.2.2 SIVA

The Speaker Identification and Verification Archives (SIVA) database consists of Italian speech generated from more than 2000 calls collected over a PSTN (public switched telephony network) network using numerous handsets in a home / office environment [29]. The SIVA corpus consists of both genders for clients and imposters alike. The system is accessed by dialling a toll free number. In the first session, 28 words which consist of digits and commands are recorded using an enumerated prompt. In the second session, the caller simply answers prompted questions (e.g. what is your name, age, etc.). In the last session, the speaker is prompted to read a continuous passage of text that resembles a concise curriculum vitae.

1.2.3 Poly Var

Poly Var is an SV corpus consisting of native and non-native speakers of French, mainly from Switzerland. It consists of about 160 hours of read and conversational speech in Swiss and French. In this database, 31 speakers made between 2 and 10 calls, while 41 speakers made more than 10 calls. In total, there are 143 speakers, 85 of whom are male. The interval between sessions ranges from days to months. These recordings were also performed in a typical home / office environment [24].

1.2.4 POLYCOST

The POLYCOST database was collected under the COST 250 (*continous speech recognition over the telephone*) European project for speaker verification. Much of the speech is non-native English with some speech in the speaker's

native language covering 13 countries in Europe. This speech was collected over international ISDN (*integrated services digital network*) telephone lines [62]. Different languages were used in order to allow researchers to investigate the effect of language on SV recognition performance. There are a total 133 speakers, 73 of whom are male. The recordings were also made in a home / office environment, and the period between successive sessions ranged from days to weeks. There were fewer than 5 sessions per speaker in total.

1.2.5 KING

The KING corpus was collectd in 1987 under a US Government research contract. The version now available from the Linguistic Data Consortioum (LDC) is referred to as KING-92, is based on a 1992 reprocessing of the original recordings. "It contains recorded speech from 51 male speakers in two versions, which differ in channel characteristics: one from a telephone handset and another from a high-quality microphone" [70]. The speakers are further subdivided into two groups of 25 and 26 speakers. These speakers were recorded from separate locations. There are 10 files for each speaker and channel, corresponding to sessions of about 30 seconds to 1 minute duration per file. The interval between consecutive sessions ranges from a week to a month, while transmission channels are a composition of clean and PSTN. These recordings were made in a sound booth.

1.2.6 YOHO

This database is designed to support text-dependent SV evaluation for secure access applications for the US government. A high-quality telephone handset (Shure XTH-383) was used to collect the speech; however, this speech was not transmitted across a telephone channel [70]. YOHO was recorded in a fairly quiet office environment with low-level office background noise (e.g. fan noise and sporadic pages over a public address system). The phrases are randomised and prompted in

a text-dependent speaker verification scenario using a *combination lock* syntax. A typical prompt could read: "Say: seventy-nine, twenty-four, ninety-seven", etc. There are 138 speakers in total, 32 of whom are female. The interval between consecutive sessions ranges from a few days to a month.

1.2.7 Switchboard I-II Including NIST Evaluation Subsets

These corpora represent some of the largest collections of recorded conversational speech available. "There are two main switchboard corpora (I and II), two phases of Switchboard-II and several subsets of Switchboard I-II used to create the NIST speaker recognition evaluation corpora" [70].

Both Switchboard-I and II were collected by a user dialling into an automated operator that connected them to another user and recorded the first 5 minutes of their conversation. This automated operator handled the information-gathering and suggested topics to be discussed by callers. The major difference between Switchboard-I and II is the demographics of the callers. In Switchboad-I, the age and location of tbe callers were diverse, whereas in Switchboard-II, the participants were obtained from certain college campuses from different parts of the US.

1.3 The Objectives of This Thesis

The first objective of this thesis is to design a baseline text-independent speaker verification architecture using concepts and methodologies detailed in contemporary literature. Once this has been accomplished, the results obtained will be compared with those of other SV architectures tested in a similar environment (NTIMIT). This is necessary in order to confirm that the design was implemented

properly.

After confirming the validity of the implementation, the next objective will be to develop and implement a robust, computationally feasible and reproducible noise compensation technique (see Chapter 4) that can be used to improve the performance of SV architectures in noisy environments. This is motivated by the fact that state-of-the-art SV systems perform exceptionally well in ideal (i.e. high SNR) environments. As a matter of fact, it is clear that there is a direct correlation between SNR and recognition performance [61]. Therefore the author believes that improving the fidelity of noise-contaminated speech can improve the performance of an SV system. Thus, stated differently, the next objective of this project is to implement a noise-suppression technique to pre-process contaminated speech in order to reduce the recognition error. **Since commercial SV systems are likely to use telephones, the author decided to use the NTIMIT database since it contains speech that has been transmitted across telephone networks.**

However, the difficulties associated with developing an SV architecture that performs perfectly in diverse acoustic environments make it very unlikely that any single research effort will solve the problem in its entirety. It is therefore the author's hope that the output of this work will provide an incremental contribution to the total solution as a whole. **In other words, the author is under no illusion that he can single-handedly solve all the problems detailed in section 1.2 within any reseanably short period of time.** Thus the ultimate objective of this work is to develop a reliable performance enhancement paradigm using established mathematical concepts that have hitherto not been applied in speaker recognition with a view to providing greater penetration into how to mitigate the problem at hand.

1.4 Scope and Limitations

In the following sub-sections, the author will outline the scope and limitations of this thesis.

1.4.1 Simulation Environment

Arguably, the best way to test an SV system is to perform live verification experiments in diverse environments and over considerable time intervals, using different handsets. This is necessary in order to capture the variability that these conditions would introduce. However, it would be virtually impossible to perform these experiments in every conceivable acoustic environment and using all available handset types. **Furthermore, the amounts of time and money required to do this are prohibitive**. Consequently, the author decided to use the NTIMIT database for all simulations related to this work.

1.4.2 Lack of Inter-Session Variability

Although NTIMIT does not cater for intersession variability owing to the contemporaneous nature of the speech files that comprise it, it does however have a few features that make it quite attractive for this study. First of all, there are a total of 630 speakers in NTIMIT, each of which has 10 utterances. These sentences were transmitted over a telephone network using both long-haul and short-haul telephone routes [70]. This undoubtedly compounds the problem since different routes would have different transfer functions, but it is likely to be the case in reality. Stated differently, how a call is actually routed may depend on availability or congestion levels in the network, making it unlikely that the same route will always be used. However, the fact that the author decided to use NTIMIT means that the problem of inter-session variability will not be addressed in this work since each user was recorded in a single session.

1.4.3 Universal Thresholds

As outlined in the next chapter, a generic SV architecture comprises (among other things) a so-called decision module, which is responsible for authenticating or refuting identity claims. To do this, a claimant's speech is parametrized and analysed for proximity with the model of the user whose identity is claimed. If the proximity is within a predefined threshold, the claim is verified, otherwise it is rejected. It is well-known that better results are obtained when each user has a unique threshold [32], as opposed to when all users are compared to a universal threshold. However, the use of speaker-specific thresholds necessitates the reservation of some data for threshold calibration purposes. Unfortunately, there is too little data in NTIMIT for some of it to be reserved for calibrating speaker-specific thresholds. Therefore it is logical that a universal threshold approach be adopted even though this would not lead to optimal performance.

1.4.4 Construction of Sentences

As mentioned in 1.4.2, each speaker in NTIMIT has 10 sentences, two of which are the same for every speaker (i.e. SA1 and SA2). These SA files are mainly intended for dialect calibration. In addition, there are 450 phonetically compact sentences (SX files) as well as 1890 phonetically diverse sentences [78]. Some SX files are the same for some speakers, whereas all SI files are different for all speakers [53]. It is thus logical to use SA and SX files in training and SI files for testing in text-independent verification.

1.4.5 Closed-set Speaker Verification

This work only addresses the problem of performing speaker verification in a closed set environment. The reason for this is that it would otherwise be difficult to isolate the cause of performance degration if an additional parameter were introduced.

1.5 Thesis Development

The remainder of this work is organized as follows:

- Chapter 2 outlines concepts and methodologies that are generally adopted in the design and implementation of state-of-the-art speaker verification architectures. These concepts and methodologies include speech parameterisation (i.e., feature extraction), speaker and anti-speaker modelling techniques, as well as decision theoretic paradigms. A fair amount of emphasis is also placed on a few evaluation metrics that are used universally to assess the performance of SV architectures.

- Chapter 3 introduces some common speech enhancement (or noise cancellation) paradigms that have been proposed by researchers in an attempt to address the problems stated in section 1.1 of this chapter. Although it is true that the performance of speaker recognition architectures depends on numerous factors as detailed in Section 1.1, only common noise cancellation or speech enhancement techniques will be explored in this chapter. The point of this is to enable the author to gain some insight into why contemporary speech enhancement techniques have not been very successful in terms of improving the performance of speaker recognition architectures in general and SV systems in particular.

- Chapter 4 provides comprehensive descriptions of both the baseline SV system and the proposed PA filter-based algorithm. The baseline system will be implemented using concepts and methodologies that are used in the design of competitive SV architectures, whereas the proposed PA filter-based design will be implemented using established mathematical and engineering concepts that have hitherto not been used in the context of speaker recognition. The designs of both architectures will be qualified using credible references.

- Chapter 5 presents the results obtained using both the baseline system as well as the proposed algorithm. These results will be compared with those obtained by other researchers using the same database (i.e. NTIMIT) and under comparable conditions (e.g. results corresponding to mismatched conditions cannot be compared with those corresponding to matches ones, for instance). The main reason for comparing the baseline results with those reported in literature is to verify the implementation. Having done that, the results corresponding to the proposed algorithm will be compared with those obtained using the baseline architecture in order to confirm the utility of the proposed algorithm.

- In chapter 6, the author will draw conclusions based on whether the proposed algorithm performed as expected, and potential ways of refining it will also be mentioned. Furthermore, a realistic analysis of contemporary SV evaluation metrics and their ability to extrapolate an SV architecture's true performance potential will be provided. Finally, the author will give a prognosis for SV architectures as a generality for the foreseeable future.

1.6 Summary

This chapter served as a brief introduction to comptemporary speaker recognition architectures and their inherent deficiencies. In particular, the author attempted to highlight the factors that negatively affect SV systems in general. Special emphasis was also placed on the databases that are used by researchers in their quest to overcome the stated problems. Lastly, an outline of this thesis was provided in an attempt to enlighten the reader about what is to follow.

Chapter 2

Overview of SV Theory

This chapter presents a comprehensive overview of contemporary speaker verification theory and also describes methodologies used in the design of state-of-the-art SV architectures. It begins with a brief description of the preliminary operations that must be performed on raw speech (i.e. the signal acquisition and quantization phases), and proceeds to illustrate the extraction of idiosyncratic parameters that individuate a speaker (i.e. the feature extraction phase). Although there are many feature extraction methods in existence, only a handful of these will be described in this chapter. Subsequently, an overview of the best-performing classification engine for text-independent speaker recognition tasks (Gaussian Mixture Model-based classifiers, used in this thesis) [41][50] is provided, followed by a discussion of decision theoretic methodologies that are used to authenticate or negate an identity claim. This is followed by a review of current SV performance metrics.

2.1 Signal Acquisition and Conditioning

Speech propagates across an air interface as a sequence of complex longitudinal acoustic waves. An acoustic-to-electrical transduction process in a microphone

converts these waves into an analogue waveform. At this stage, the analogue signal might be conditioned with an anti-aliasing filter to limit the bandwidth to the Nyquist rate [21] and quantized using a suitable resolution. A 16-bit resolution is common in most databases, while sampling rates normally range from 8 KHz to 20 KHz. It is also common practice to partition speech into frames in order to make the computation of features more efficient.

The next section describes how features may be extracted from digitized speech data.

2.2 Feature Extraction

In the context of speaker recognition, feature extraction is a process in which speaker-specific information is derived from a speech waveform. This contrasts with speech recognition, whose main goal is to derive lexical information from the signal regardless of who the originator of it might be. Surprisingly, however, a lot of feature extraction techniques have been used successfully for both speech and speaker recognition without any modification.

A comprehensive description of contemporary feature extraction techniques is given in the following subsections.

2.2.1 Linear Predictive Coding (LPC)

Linear predictive coding was initially conceptualised as method for representing speech signals [45][1] . It attempts to model the vocal tract as a parametric entity which is described mathematically by a transfer function $H(z)$ that alters the spectral content of an acoustic wave as it passes through it [47]. Ideally, this transfer

[1]Much of the information in this subsection was obtained from this reference as well as [21].

function should consist of poles and zeros. However, if only voiced speech segments are used, an all-pole model of $H(z)$ is adequate.

The all-pole LP analysis models a signal s_n as a linear combination of its past values and a scaled present input

$$s_n = -\sum_{k=1}^{p} a_k . s_{n-k} + G.u_n \tag{2.1}$$

where s_n is the present output, p is the prediction order, a_k are the model parameters known as predictor coefficients (PCs), s_{n-k} are past outputs, G is a gain scaling factor, and u_n is the present input. In speech applications, the input u_n is generally ignored [21]. Therefore, the LP approximation $\widehat{s_n}$, depending only on past samples, is

$$\widehat{s_n} = -\sum_{k=1}^{p} a_k . s_{n-k} \tag{2.2}$$

This significantly simplifies the problem of estimating a_k because the source (i.e. the glottal input) and filter (i.e. the vocal tract) have been decoupled. However, this means that the source u_n, which corresponds to the vocal tract excitation, is not modelled by these PCs. Consequently, it is reasonable to deduce that some speaker-dependent characteristics might be present in this excitation signal. Therefore, since the excitation signal is ignored, it is fairly probable that important speaker-discriminative information is lost.

An observation of (2.1) and (2.2) reveals that the prediction error (also known as the residual) is given by

$$e_n = s_n - \widehat{s_n} = Gu_n \tag{2.3}$$

Thus the prediction error is identical to the scaled input signal $G.u_n$. The LP transfer function $H(z)$ is written from (2.1) as

$$H(z) \equiv \frac{S(z)}{U(z)} = \frac{Z[s_n]}{Z[u_n]} \tag{2.4}$$

which produces

$$H(z) = \frac{G}{1+\sum_{k=1}^{p} a_k z^{-k}} \equiv \frac{G}{A(z)} \tag{2.5}$$

where $A(z)$ is known as the pth-order inverse filter. LP analysis determines the predictor coefficients of the inverse filter $A(z)$ that minimise the prediction error e_n in a sense. Not surprisingly, several derivatives of LPC were developed. These derivatives include LPC-based cepstral coefficients and line spectral pair (LSP) coefficients and are described in subsequent subsections.

2.2.2 Cepstral Coefficients

In 1963, Bogert et al published a paper with the title "The Quefrency Analysis of Time Series for Echoes" [7][2]. They noted that the logarithm of a signal's power spectrum containing an echo has an additive periodic component due to the echo. Thus they determined that the Fourier transform of the logarithm of the power spectrum should exhibit a peak at the echo delay. They called this function the cepstrum, swopping letters in the word "spectrum".

There are two common methods used for the computation of cepstral features. The first is the direct computation method, based on the power spectrum of the signal $x(t)$ derived from a Fourier Analysis.

[2]Virtually all the information in this subsection was obtained from this reference.

$$x(t) \rightarrow \boxed{\text{FFT}} \rightarrow \boxed{\text{Mel}} \rightarrow \boxed{\text{Log}} \rightarrow \boxed{\text{DCT}} \rightarrow c(n)\ [15] \tag{2.6}$$

In the first method, a raw speech signal is initially transformed using FFT. Subsequently, the spectral coefficients are combined to log-energies in uniformly spaced filter bands on a Mel frequency scale. The Mel transformation de-emphasises high frequencies based on the non-linear perception of the frequency of sounds in human beings. Finally, the discrete cosine transform (DCT) is used to convert the log Mel spectrum into the cepstral domain. The first cepstral coefficient C_0 describes the overall energy contained in the spectrum, while C_1 measures the balance between the upper and lower halves of the spectrum. The second approach is based on a linear predictive coding (LPC) scheme, and is described next.

Let S be a sampled speech waveform. The linear prediction scheme uses samples $n-1$ to $n-p$ to predict the nth sample. That is,

$$\widehat{s_n} \simeq \sum_{i=1}^{p} a_i . s_{n-i} \tag{2.7}$$

As explained in section 2.2.1, minimisation of the mean squared prediction error delivers the linear prediction coefficients $\{a_1, \ldots, a_p\}$. The cepstral coefficients can then be derived efficiently from the LPC coefficients by way of a simple recursive algorithm as follows:

$$c_i = a_i + \tfrac{1}{i} \sum_{j=1}^{i-1} (j) a_{i-j} c_j \bigvee \mathrm{i} = 1, \ldots, n \tag{2.8}$$

The zeroth cepstral coefficient c_o cannot be calculated directly using LPC analysis, but because it represents the overall energy in the spectrum, the LPC cepstral coefficients are usually augmented with log-energy to replace c_0. Several other concepts have been proposed in order to transform LPC coefficients into a set of orthogonal parameters which should be independent of the linguistic informa-

tion in an utterance, yet highly indicative of the speaker. However, none of these techniques are still used today.

2.2.3 Parametric Feature Sets (PFS)

Parametric feature sets are similar to the MFCCs described in the previous subsection except that their generation does not require a Mel-scale filterbank [17]. In the generation of PFS, the pitch component is removed in the spectral domain prior to translation to the cepstral domain (i.e. liftering). This liftering process removes the pitch component by liftering the spectrum using a 41-point finite impulse response (FIR) filter. These features include spectral compression and filtering.

For a particular pair of parameters α and β, the spectrum is sampled non-linearly such that

$$A \sum_{i=1}^{\alpha} \beta^{i-1} = N/2 \tag{2.9}$$

where N is the size of a DFT spectrum, A is a constant greater than 1, and α and β describe spectral compression. It is thus possible to fine-tune the performance of a PFS-based recognition system by choosing specific permutations of α and β. Typical values for α could be 4, 6, 8, 10, 12, etc, while β might be one of 1.0, 1.2, 1.7, 2.0, etc. The values of α and β that were chosen for this work are 4 and 1.6 respectively [58]. These feature sets were investigated by Mashao [54] in his PhD thesis.

2.2.4 Line Spectral Pairs (LSPs)

LSPs are a representation of the PCs of the inverse filter $A(z)$ in equation 2.5, "where the p zeros of $P(z)$ are mapped onto the unit circle in the z-plane using a

pair of auxiliary $(p+1)$-order polynomials: $P(z)$ (symmetric) and $Q(z)$ (asymmetric)" [21]. That is,

$$A(z) = \tfrac{1}{2}[P(z) + Q(z)]\,, \tag{2.10}$$

where

$$P(z) = A(z) + z^{-(p+1)}A(z^{-1}) \tag{2.11}$$

and

$$Q(z) = A(z) - z^{-(p+1)}A(z^{-1}) \tag{2.12}$$

where the LSPs are the frequencies of the zeros of $P(z)$ and $Q(z)$. By definition, an LP synthesis filter (i.e. a stable one) should have all its poles inside the unit circle in the z-plane. The resultant inverse filter is therefore minimum phase inverse since it has no poles or zeros outside the unit circle. "Any minimum phase polynomial can be mapped by this transform to represent each of its roots by a pair of frequencies (phases)" [21] with a magnitude of unity.

Owing to the fact that the PCs are real, *the Fundamental Theorem of Algebra* guarantees that the roots of $A(z)$, $P(z)$, and $Q(z)$, will occur in conjugate pairs. Due to this property, the bottom half of the z-plane is redundant. The LSPs at 0 and π are always present by virtue of how P and Q are constructed. Therefore, the PCs can be represented by the number of LSPs equal to the prediction order p and are described by the frequencies of P and Q in the top half z-plane.

These LSPs are subject to the constraint:

$$0 = \omega_0^{(Q)} < \omega_1^{(P)} < \omega_2^{(Q)} < \cdots < \omega_{p-1}^{(P)} < \omega_p^{(Q)} < \omega_{p+1}^{(P)} = \pi. \tag{2.13}$$

Each zero of $A(z)$ maps into one zero in each $P(z)$ and $Q(z)$. When the $P(z)$ and $Q(z)$ frequencies are close, it is likely that the original $A(z)$ zero was close to the unit circle, and a formant is thus likely to be located between the corresponding LSPs. Distant P and Q zeros are likely to correspond to wide bandwidth zeros of $A(z)$ and most likely contribute only to shaping or spectral tilt.

2.2.5 Maximum Auto-Correlation Values (MACVs)

"In MFCC features, only the system part of the speech signal is effectively utilised" [73][3]. There are two ways in which pitch information may be utilised. The first uses a dedicated pitch-based verification module and fuses its output with that of a generic SV system prior to reaching the final accept or reject decision. The front-end for the dedicated module may consist of a voiced / unvoiced frame detector, followed by a pitch frequency extractor.

The second technique entails incorporating pitch information directly into the feature vector. The pitch period may be detected by using the autocorrelation function, which for a speech frame $\vec{s}^T = [s_i]_{i=1}^{N_s}$ is defined as:

$$R(k) = \frac{1}{N_s} \sum_{i=1}^{N_s-k} s_i s_{i+k} \bigvee k = 0, 1, ..., N_s - 1 \tag{2.14}$$

If $\vec{s}$ is periodic and has a period of P samples, then $\{R(k)\}_{k=0}^{N_s-1}$ will show a peak at a lag equal to P. The pitch frequency is typically between $60 - 160$ Hz for males and $160 - 400$ Hz for females, implying that valid pitch lags are roughly between 2.5 ms and 16 ms. Thus the period of $\vec{s}$ can be found by locating the maximum value of $\{R(k)\}_{k=0}^{N_s-1}$ in the 2.5 ms to 16 ms range. This method thus permits the recovery of the pitch period when using a telephone channel since it

[3]The information provided in this subsection was derived mainly from this reference.

limits the bandwidth of speech signals between 300 and 3400 Hz.

Regrettably, the autocorrelation method is susceptible to pitch halving and doubling, among other things. To illustrate this, suppose that a signal is periodic with period P. According to harmonic analysis, the signal will also be periodic with period $2P$, $3P$, etc. Hence $\{R(k)\}_{k=0}^{N_s-1}$ will also have maxima at lags equal to $2P$, $3P$, etc. Moreover, one of the spurious maxima in the signal may be a global maximum. Therefore it is probable that the pitch period will be identified as $2P$, which is referred to as pitch halving. In cases where the Mth formant dominates the signal's energy (which could well occur if a telephone channel is used), there would be a maximum at a lag of P/M. Hence the pitch period would be identified as $P/2$, which is known as pitch doubling.

However, if the speech frame is unvoiced, both these pitch extraction techniques provide random values, indicating that their output cannot be incorporated into the feature vector for each frame. The MACV feature set addresses these problems by extracting pitch information from the auto-correlation function instead of attempting to determine it directly. A formal description of the MACV algorithm is as follows[4]:

1. Compute the auto-correlation function $\{R(k)\}_{k=0}^{N_S-1}$.

2. Normalise $\{R(k)\}_{k=0}^{N_S-1}$ by its maximum, i.e.,

$$\{\hat{R}(k)\}_{k=0}^{N_s-1} = \left(\frac{R(k)}{R(0)}\right)_{k=0}^{N_s-1}$$

3. Divide the higher portion (from 2.5 ms to 16 ms) of $\{R(k)\}_{k=0}^{N_S-1}$ into N_M equal parts (typically, $N_M = 5$).

[4]This MACV feature extraction description was taken verbatim from [73]

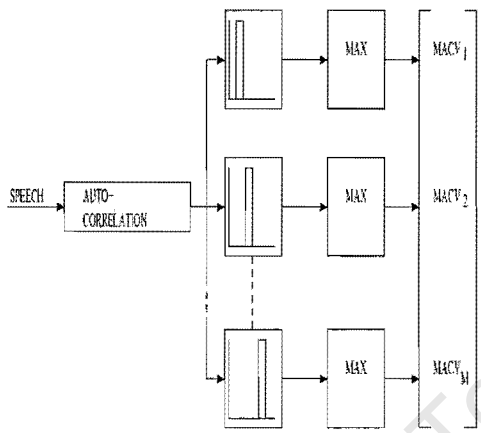


Figure 2.1: MACV feature extraction [73]

4. Find the maximum value of each of the N_M parts.

5. The N_M Maximum Auto-Correlation Values (MACVs) form an N_M-dimensional feature vector.

This procedure is summarised graphically in Figure 2.1.

2.3 Speaker Modelling Using GMM

For text-independent speaker verification, "the most successful likelihood function has been Gaussian mixture models" [71][5]. In text-dependent applications, temporal information in a speech signal may be incorporated by using hidden Markov models (HMMs) as the basis for the likelihood function. Nevertheless,

[5]Much of the information in this subsection was derived from this reference.

the use of more complicated likelihood functions such as those based on HMMs have not proven superior to GMMs for text-independent SV tasks.

For a D-dimensional feature vector x, the mixture density used for the likelihood function is defined as

$$P(X|\lambda) = \sum_{i=1}^{M} w_i P_i(x) \tag{2.15}$$

This density is a scaled linear combination of M unimodal Gaussian densities, $P_i(x)$, each parameterised by a $D \times 1$ vector, μ_i (i.e. the mean vector), and a $D \times D$ covariance matrix, $\sum_i$. That is,

$$P_i(\mathbf{x}) = \frac{1}{(2\pi)^{D/2}|\sum_i|^{\frac{1}{2}}} e^{\{-\frac{1}{2}(\mathbf{x}-\mu_i)^T(\mathbf{x}-\mu_i)\}} \tag{2.16}$$

The mixture weights, w_i, satisfy the constraint $\sum_{i=1} w_i = 1$. The parameters of the density model are described by $\lambda = \{w_i, \mu_i, \sum_i\}$, where $i = 1, \ldots, M$. Although the general model form supports full covariance matrices, a diagonal covariance matrix may also be used. This could be done for several reasons. Firstly, the density modelling of an Mth-order full covariance GMM can equally well be achieved using a larger-order diagonal covariance GMM. Secondly, diagonal-matrix GMMs are more computationally efficient than full covariance GMMs for training since repeated inversions of a $D \times D$ matrix are not required. Thirdly, it has been observed that diagonal matrix GMMs outperform full matrix GMMs.

2.4 Anti-Speaker Modelling

Speaker verification does not only require a model of who the speaker is, but also a model describing other speakers [46][6]. This is because the likelihood statistic $P(S_i|X)$, which determines the probability of speaker S_i given the acoustic observations X, cannot be computed directly. However, Bayes' theorem may be used to rewrite the likelihood as

$$P(S_i|X) = \frac{P(X|S_i)P(S_i)}{P(X)} \tag{2.17}$$

But this theorem introduces two new prior probabilities that cannot be expressed directly: $P(S_i)$, the probability that speaker S_i is present, and $P(X)$, the probability that these observations will occur. Assuming that $P(S_i)$ is the same for every speaker so that $P(S_i) = P_s \; \forall i \in I$, where I is the set of all possible speakers, and writing $P(X)$ as a sum of conditional probabilities, (2.17) may be rewritten:

$$P(S_i|X) = \frac{P(X|S_i)P_s}{\sum_{j \in I} P(X|S_j)P(Sj)} = \frac{P(X|S_i)}{\sum_{j \in I} P(X|S_j)}. \tag{2.18}$$

All terms in (2.18) can now be determined either explicitly or through simplifying assumptions. However, evaluating the complete set I of all possible speakers is not tractable. There are two popular techniques that are normally used to approximate the denominator of (2.18), namely, cohort modelling and world modelling [23]. Cohort modelling estimates the set I by a finite set of speakers F_i who sound like speaker S_i. On the other hand, world modelling reduces the set I to a size of one, containing only a single hypothetical speaker S whose model is trained using speech from many different speakers who represent the 'world' of possible speakers. The cohort method approximates equation (2.18) as:

[6]Most of the information in this section was taken from this reference.

$$P(S_i|X) \overset{cohort}{\approx} \frac{P(X|S_i)}{\sum_{j \in F_i} P(X|S_j)}, \tag{2.19}$$

whereas the world model-based approximation yields

$$P(S_i|X) \overset{world}{\approx} \frac{P(X|S_i)}{P(X|S)}. \tag{2.20}$$

It is reported that the world model approach out-performs the cohort approach for small (i.e. below about 20) cohort sizes [22]. In addition, the world model approach is more attractive computationally since only one anti-speaker score ought to be evaluated during testing. Not surprisingly, the world model approach is more commonly used, and it comes in different flavours [69]. According to this reference, these flavours include completely speaker-independent implementations, as well as gender- or handset-dependent implementations where the choice for the world model depends on the gender of the claimed identity or the type of handset used by the speaker.

An alternative approach for estimating the imposter model is the so-called universal background model, or UBM. In this approach, pooled training data from all speakerss is used to create a large mixture model consisting of many Gaussians. "The UBM is a large GMM trained to represent the speaker-independent distribution of features" [68]. Specifically, it is used for selecting speech that is "reflective of the expected alternative speech to be encountered during recognition" [68]. This applies to both the type and quality of speech, as well as the composition of speakers. For instance, in the NIST SRE (speaker recognition experiments) single-speaker verification tests, it is known beforehand that the speech comes from local and long-distance telephone calls and that male hypothesized speakers will only be tested against male speech. If the gender composition of the alternative speakers were unknown, a UBM model would be trained using gender-independent speech.

2.5 Distance Measures

Distance measures refer to techniques of calculating the proximity between parameter vectors. Typically, one of the vectors is calculated from speech of the unknown speaker while the other vector is calculated from that of a known speaker. However, "some pattern-matching techniques require that vectors from the same speaker be compared to each other to determine the expected variance of the speaker in question" [45][7]. Some of the most common distance measures are described in the following subsections.

2.5.1 Euclidean Distance

The Euclidean distance measure between two feature vectors is calculated by

$$d(a,b) = (\sum_{i=1}^{p}(a_i - b_i)^2)^{\frac{1}{2}} \tag{2.21}$$

where a_i and b_i are the ith components of the two vectors to be compared and p is the number of vectors to compare.

2.5.2 Manhattan Distance

On the other hand, the Manhattan distance measure between two vectors is given by

$$d(a,b) = \sum_{i=1}^{p} |a_i - b_i|. \tag{2.22}$$

However, it is worth noting that the Euclidean and Manhattan distance measures are not suitable for comparing two vectors of LPC coefficients because of the

[7]The information in sub-sections 2.5.1 to 2.5.3 was taken from this reference.

features' inter-dependence. There is a special distance measure that may be used for LPC coefficients, and it is described next.

2.5.3 Likelihood Ratio Distortion[8]

The likelihood ratio distortion is defined as:

$$d_{LR}(a,b) = \frac{\mathbf{b}^T \mathbf{R}_a \mathbf{b}}{\mathbf{a}^T \mathbf{R}_a \mathbf{a}} - 1 \tag{2.23}$$

from which the log likelihood distance is then computed simply as:

$$d_{LLR} = log(d_{LR}) \tag{2.24}$$

where **a** and **b** are vectors of LPC predictor coefficients, $\mathbf{R}_a$ is the Toeplitz auto-correlation matrix (a bi-product of the calculation of the PCs) associated with **a**, and T signifies transposition.

2.5.4 Divergence Measure

Divergence is an information theory-derived distance measure that is used to compute the proximity between two classes [21][9]. It provides a theoretical framework for ranking features and evaluating class discriminability.

Suppose the likelihood of occurence of observation o that belongs to class c_i is:

$$p_i(o) = p(o|c_i) \tag{2.25}$$

[8]This distance measure only applies to LPC coefficients

[9]Much of the information in this sub-section and the next were derived from this reference.

and

$$p_j(o) = p(o|c_j) \tag{2.26}$$

for class c_j. Thus the *discriminating information* of pattern o for class c_i versus class c_j is:

$$u_{ij} = ln\ \frac{p_i(o)}{p_j(o)}. \tag{2.27}$$

The average discriminating information for class ω_i is defined as:

$$I(i,j) = \int_o p_i(o) \ln \frac{p_i(o)}{p_j(o)} do, \tag{2.28}$$

while the *divergence*, defined as the total average information for discriminating class ω_i from ω_j, is calculated using

$$J_{i,j} = I(i,j) + I(j,i) = \int_o [p_i(o) - p_j(o)] ln \frac{p_i(o)}{p_j(o)} do. \tag{2.29}$$

A detailed account of how to select features with this measure may be found in [21].

2.5.5 Log Area Ratios (LARs)

"The human vocal tract can be modelled as an electrical transmission line, a waveguide, or an analogous series of cylindrical tubes" [21]. At each junction, there can be a mismatch in impedance or, analogously, a difference in cross-sectional areas between tubes. At each boundary, a portion of the acoustic wave is

transmitted and the remainder is reflected (assuming lossless tubes). The reflection coefficients k_i represent the proportion of reflected waves at these discontinuities. Assuming that the acoustic tubes are of equal length, then the time required for sound to propagate through each tube is constant. This assumption enables simple z transformation for digital filter simulation. For instance, a series of five acoustic tubes of equal lengths with cross-sectional areas A_1, $A_2, \cdots$, A_5 could represent a fourth-order system that might fit a vocal tract excluding the nasal cavity. Given boundary conditions, the reflection coefficients are determined by the ratios of adjacent cross-sectional areas. For a pth-order system, the boundary conditions given in (2.30) correspond to a closed glottis (zero area) and a large opening following the lips. That is,

$$\begin{aligned} &A_0 = 0 \\ &A_{p+1} \gg A_p \\ &k_i = \tfrac{A_{i+1}-A_i}{A_{i+1}+A_i} \bigvee i = 1, 2, \cdots, p \end{aligned} \tag{2.30}$$

Thus, the reflection coefficients may be derived from a tube model or an autoregressive model.

2.6 Decision Theory

In speaker verification, speaker S is accepted as the claimed speaker S_c if

$$P(S_c|X) > P(\overline{S_c}|X) \tag{2.31}$$

where $\overline{S_c}$ represents the set of all possible rival speakers, and the right hand side is the probability of the speaker being anyone else but S_c [20][10]. Typically, this

[10]The info in this section was mainly derived from this reference.

is stated with some margin or threshold. That is, a speaker S is accepted as the claimed speaker S_c if

$$\frac{p(S_c|X)}{p(\overline{S_c}|X)} > \delta_c \tag{2.32}$$

where $\triangle_c$ is a threshold (>1) which must be taylored uniquely for each potential customer S_c to ensure optimal performance. Taking logarithms on both sides of inequality (2.31) enables the acceptance criterion to be rewritten as:

$$log\ P(X|S_c) - log\ P(X|\overline{S_c}) > \triangle_c \tag{2.33}$$

Expression (2.33) is otherwise known as the likelihood ratio criterion. In general, a large enough value of this difference suggests that the identity of S_c is validated, whereas a lower value normally implies that the claimant is an impostor. If a UBM model is used, the acceptance criterion becomes

$$log\ P(X|S_c) - log\ P(X|S_{UBM}) > \triangle_c. \tag{2.34}$$

However, it is also possible for speakers to be compared against a universal threshold if there is inadequate training data. Nevertheless, this approach leads to a sub-optimal recognition performance.

2.7 Performance Metrics

Since speaker verification system is a two-class decision task, it follows that the system can make two types of errors [20][11]. The first type of error is a false acceptance (FA), and it represents the proportion of mistakenly accepted impostor

[11]Most of the information in this section was derived from this reference.

claims. The second error is a false rejection (FR), where a valid client is mistaken for an impostor. Thus the performance of an SV architecture is specified in terms of the false acceptance rate (FAR) and the false rejection rate (FRR). These errors are calculated as follows:

$$\text{FAR } [\%] = \frac{I_A}{I_T} \times 100 \tag{2.35}$$

and

$$\text{FRR } [\%] = \frac{C_R}{C_T} \times 100 \tag{2.36}$$

where I_A is the number of accepted impostors, I_T is the total number of impostor attacks, C_R is the number of rejected true claimants, and C_T is the total number of true claimant classification tests.

Since these errors depend on the same decision threshold, reducing the FAR increases the FRR, and vice versa. The trade-off between these errors is adjusted using the decision threshold $\triangle_c$ in (2.33). Depending on the application, more emphasis may be placed on one error over the other. For instance, in a high security environment, it may be preferable to have FAR as low as possible, even at the expense of a slightly higher FRR.

The trade-off between FAR and FRR can be represented graphically using the so-called receiver operating characteristics (ROC) plot, or a detection[12] error trade-off (DET). The ROC is a plot on a linear scale, while the DET is on a quasi-log scale. In both cases the FRR is plotted as a function of the FAR.

[12]Note that speaker verification is sometimes termed speaker detection.

To express the performance of an SV system using a single metric, the equal error rate (EER) is often used. In this case, the system is configured to operate with FAR = FRR. This is done by adjusting the threshold in small increments until the errors converge to a common value. It must be noted that the threshold is adjusted to obtain desired performance on *test* data (i.e., data hitherto unseen by the system). Such a threshold is known as an *a posteriori* threshold. On the other hand, if the threshold is fixed before finding the performance, the threshold is known as an *a priori* threshold. The *a priori* threshold can be found empirically using training data or *evaluation* data (i.e. data hitherto unseen by the system, but separate from *test* data).

In addition, the performance of an SV system (or any other two-class discrimination problem) may be expressed indirectly using a metric known as the D'. The abbreviation D' stands for the discriminability index [3]. Roughly stated, the D' is a measure of the extent of separability of two distributions.

Figure 2.2 shows two pairs of distributions with different D' values. As can be seen, a low D' value (i.e. 1 in Figure 2.2) means that there is considerable overlap for the first pair of distributions (i.e. high recognition rates), whereas a high D' value (i.e. D' = 3 in the figure) shows much less overlap (and hence much lower recognition rates). In the context of SV, these distributions correspond to impostor and client scores. Therefore a low D' value suggests that a lot of clients will be rejected as impostors. Conversely, it also means that many impostors will be accepted as legitimate clients.

Mathematically, the D' is calculated as:

$$D' = \frac{\mu_A - \mu_B}{\sqrt{\frac{\sigma_A^2 + \sigma_B^2}{2}}} \tag{2.37}$$

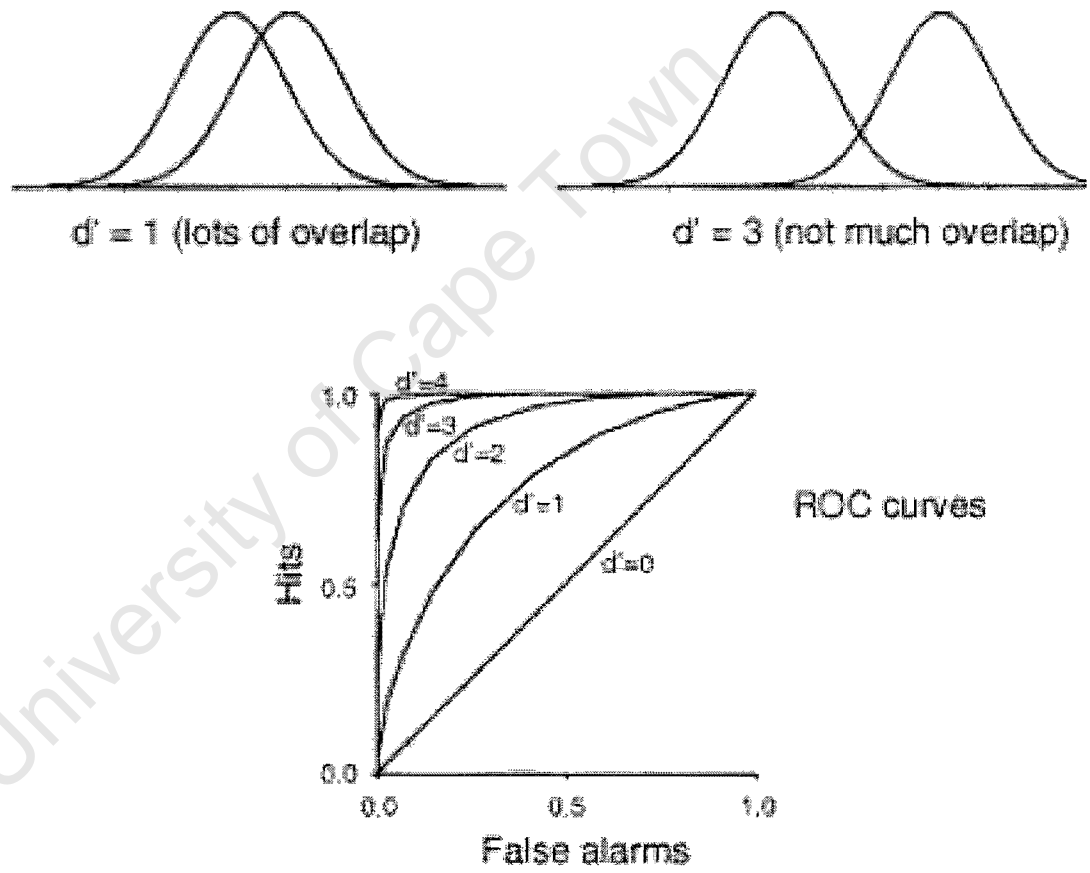


Figure 2.2: Comparing two pairs of distributions with different D' values [5].

where μ_A refers to the mean of 'distribution A' and σ_A^2 is the associated variance. An easy way to understand the D' is to think of it simply as the $separation/spread$, where $separation$ refers to the distance between the two means and spread is the average standard deviation [5].

2.8 Summary

In this chapter, the author presented a detailed overview of concepts and techniques that are used to design contemporary SV architectures. Firstly, popular feature extraction techniques were presented. Thereafter, speaker and anti-speaker modelling theory was covered. Subsequently, common distance measures were described, and, finally, performance metrics were presented.

Chapter 3

Noise Cancellation Techniques

"The primary objective of any enhancement method in the context of speech processing is to reduce the effect of any signal that is alien to and disruptive of the message conveyed among participants in a communicative event (whether it be human or automatic speech recognition machines)" [65].

This chapter provides a thorough review of common speech enhancement and noise cancellation techniques that are (or could be) used to improve the performance of speaker recognition architectures in real-world conditions. It is divided into two main parts, the first of which deals with linear filtering techniques, while the second deals with non-linear filtering techniques.

These noise cancellation paradigms can further be divided into two broad categories, namely, single-channel and multi-channel enhancement techniques [61]. As the names imply, single-channel enhancement techniques are only applicable in cases where there is one acquisition channel (e.g. a telephone channel), whereas multi-channel enhancement techniques are applicable only when there are several acquisition channels, (e.g a microphone array configuration). Since this thesis seeks to improve the performance of SV architectures for telephony speech (i.e.

using NTIMIT), only single-channel enhancement concepts will be considered.

3.1 Linear Filtering Techniques

In this section, common speech enhancement techniques are discussed. However, it is important to note that some of these concepts are still called "speech enhancement" techniques even though their application is not necessarily restricted to speech processing (e.g. spectral subtraction). In fact, a lot of techniques stated in this section have also been applied to image processing.

3.1.1 Spectral Subtraction

"The spectral subtraction method is the most suitable technique for the elimination of stationary noise from a degraded speech signal" [63]. Consider a speech signal s that has been corrupted by additive noise n resulting in a noisy speech signal $s + n$. Using a voice activity detector (VAD), it is possible to estimate the noise amplitude spectrum, $|N_j(e^{i\Theta})|$, which can then be used to estimate the original clean speech signal amplitude spectrum as follows:

$$|\widehat{S_j}(e^{i\Theta})| = |S_j(e^{i\Theta}) + N_j(e^{i\Theta})| - |\widehat{N_j}(e^{i\Theta})| \tag{3.1}$$

where the addition term represents the amplitude spectrum of the observed noisy speech signal and $|\widehat{S_j}(e^{i\Theta})|$ is the clean speech signal amplitude estimation and $|\widehat{N_j}(e^{i\Theta})|$ is the noise estimate[61]. It is assumed that the phase of the signal is the same as that of the signal plus noise. However, it is possible to over-estimate the noise amplitude term in equation (3.1), thus giving rise to negative values of $|\widehat{S_j}(e^{i\Theta})|$. A simple solution is to set all such negative values to 0. Unfortunately this simplistic solution introduces spectral spikes that give rise to so-called musical noise. The remedy is explained in the following sub-section.

3.1.2 Spectral Subtraction with Over-Subtraction

This technique was introduced in order to compensate for the "musical noise" phenomenon. In contrast to the classical spectral subtraction algorithm described above, spectral subtraction with over-subtraction estimates the clean speech spectrum magnitude as:

$$|\widehat{S_j}(e^{i\Theta})| = |S_j(e^{i\Theta}) + N_j(e^{i\Theta})| - \alpha|\widehat{N_j}(e^{i\Theta})| \tag{3.2}$$

if

$$|S_j(e^{i\Theta}) + N_j(e^{i\Theta})| - |\widehat{N_j}(e^{i\Theta})| > \beta|\widehat{N_j}(e^{i\Theta})| \tag{3.3}$$

otherwise

$$|\widehat{S_j}(e^{i\Theta})| = \beta|\widehat{N_j}(e^{i\Theta})| \tag{3.4}$$

where $\alpha > 1$ reduces the occurance of negative values that give rise to spectral spikes, and "$0 < \beta \ll 1$ sets a spectral flooring that reduces the perception of musical noise" [61].

3.1.3 Wiener Filtering

"The goal of Wiener filtering is to obtain an estimate of the original signal from a degraded version of the signal" [13][1]. Assuming that the original signal $s(t)$ was corrupted by an additive noise function $n(t)$, then the degraded signal $x(t)$ is represented by:

[1] The information in this sub-section was mainly derived from this reference.

$$x(t) = s(t) + n(t). \tag{3.5}$$

The parameters of the Wiener filter are computed such that the expectation of the error (i.e. the difference between the clean speech signal $s(t)$ and its estimate, $\widehat{s(t)}$,) is minimised. That is, the Wiener filter seeks to minimise the expression $E|s(t) - \widehat{s(t)}|^2$.

Assuming that $s(t)$ and $\widehat{s(t)}$ have zero means and that they are jointly stationary, minimising the error yields:

$$H(\omega) = \frac{P_{xs}(\omega)}{P_{xx}(\omega)} \tag{3.6}$$

where $P_{xs}(\omega)$ is the power spectrum of the cross-correlation between $s(t)$ and $x(t)$, $P_{xx}(\omega)$ is the power spectrum of $x(t)$ and $H(\omega)$ is the Wiener filter transfer function. If the noise is additive and $n(t)$ and $s(t)$ are uncorrelated, equation (3.6) may be rewritten as:

$$H(\omega) = \frac{P_{ss}(\omega)}{P_{ss}(\omega)+P_{nn}(\omega)} \tag{3.7}$$

where $P_{ss}(\omega)$ is the power spectrum of the clean speech signal and $P_{nn}(\omega)$ is the power spectrum of noise. The Wiener filter works by multiplying the value of each frequency component by a factor between 0 and 1, which is proportional to an estimate of the signal-to-noise ratio. Thus heavily contaminated regions will be suppressed (i.e. factor close to 0), whereas high SNR regions will pass through almost transparently (i.e. factor close to 1). An estimate $\widehat{s(t)}$ of the clean speech is then derived simply as:

$$\widehat{s(t)} = x(t) * h(t) \tag{3.8}$$

where $h(t)$ represents the Wiener filter impulse response, and * denotes convolution.

However, an observation of (3.7) reveals that it is necessary to have an estimate of the clean speech signal, $s(t)$. This may be derived using the spectral subtraction or any other appropriate technique. Furthermore, $P_{ss}(\omega)$ may be estimated as

$$\hat{P}_{ss}(\omega) = P_{xx}(\omega) - P_{nn}(\omega). \tag{3.9}$$

But this method does not give a good estimate of the original signal power spectrum, so an iterative method is used in which the Wiener filter transfer function is initially represented as $\frac{P_{xx}(\omega)-P_{nn}(\omega)}{P_{xx}(\omega)}$, and then the noisy signal $x(t)$ is filtered to give a new estimate of the original signal. This new estimate is then substituted into equation (3.7), and the procedure is repeated until convergence is reached.

3.1.4 Cepstral Mean Normalisation

Channel characteristics usually differ from one session to another, making it difficult to execute reliable speech recognition transactions over extended time periods [16]. A possible method of compensating for this intersession variabily is cepstral mean normalisation (CMN). CMN essentially calculates the cepstral mean, calculated across the entire utterance, and subtractes it from each frame.

As stated in section 1.1.2, speech that has been transmitted across a telephone channel is convolved with the channel's impulse response. In the log cepstral domain, this convolution is equivalent to a simple addition which can be rectified by subtracting the cepstral mean from all input vectors [1]. In practice, however, the mean is normally computed over limited quantities of speech data, hence it is not totally accurate. In any case, CMN is still a very effective technique in

practice as it compensates for long-term spectral effects (e.g. those caused by different microphone types and transmission channels [1]).

.

3.2 Non-linear Filtering Techniques

It is well known that a signal that has been contaminated by additive noise can be denoised quite effectively using a simple linear filtering technique (e.g. a Wiener filter or spectral subtraction). This is because a linear filter is ideal under the mean square error criterion for an additive Gaussian noise process [6]. However, telephony speech is known to suffer distortion as a result of different noise processes, including non-Gaussian noise, such as impulsive noise [43][56][39][4][35].

Moreover, non-Gaussian noise is generally "neglected within the system design philosophy for reasons of complexity and tractability" [36]. In other words, non-additive or non-Gaussian noise processes are generally not considered during the design of speech enhancement techniques because they are difficult to model and analyse. Furthermore, linear filters are known to perform poorly in the presence of non-additive or non-Gaussian noise [34].

By contrast, non-linear filters are more robust than linear ones [6]. A filter is robust if "deviations from the statistical assumptions for which it is optimal in one way or another do not greatly affect its performance" [6]. **It is therefore logical to infer that non-linear filtering techniques could be more effective at denoising speech that has been transmitted across a telephony network since it is perturbed by heterogeneous noise processes.**

These non-linear filters may be implemented using neural networks, polynomials (e.g. Volterra or Taylor series expansions), or other function approximation tech-

niques [77]. The following sub-sections present an overview of some non-linear filtering techniques that could find application in speech processing (in the context of speaker recognition, that is). A complete appraisal of non-linear filters may be found in [48].

3.2.1 Non-linear Spectral Subtraction

In non-linear spectral subtraction, the noise can be estimated as

$$|\widehat{N}_j(e^{i\Theta})| = \lambda_N|\widehat{N}_{j-1}(e^{i\Theta})| + (1-\lambda_N)|\widehat{N}_j(e^{i\Theta})| \tag{3.10}$$

and

$$\begin{aligned}|S_j(e^{i\Theta}) + N_j(e^{i\Theta})| = \lambda_{N+S}|S_{j-1}(e^{i\Theta}) + N_{j-1}(e^{i\Theta})| + \\ (1-\lambda_{N+S})|S_j(e^{i\Theta})+N_j(e^{i\Theta})|\end{aligned} \tag{3.11}$$

where λ_N is the so-called extended noise model and λ_{N+S} is a model of the noisy speech signal [61][2]. For the extended noise model, a generic function $\Phi[\rho_j(e^{i\Theta}), \alpha_j(e^{i\Theta}), |\widehat{N}_j(e^{i\Theta})|]$ is used that depends on the noise estimator $N_j(e^{i\theta})$, the spectral over-subtraction factor, $\alpha_j(e^{i\Theta})$, and the signal-to-noise ratio of each spectral component of the analysis frame, $\rho_i(e^{i\Theta})$. Function Φ is an arbitrary non-linear function that incorporates the subtraction process and takes into account the signal-to-noise ratio of each spectral component. It is bounded as follows:

$$|\widehat{N}_j(e^{i\Theta})| \leq \Phi[\rho_j(e^{i\Theta}), \alpha_j(e^{i\Theta}), |\widehat{N}_j(e^{i\Theta})|] \leq 3|\widehat{N}_j(e^{i\Theta})]. \tag{3.12}$$

[2]Most of the information in this sub-section was derived from this reference.

3.2.2 Volterra filters

The Volterra series expansion is a popular and widely used method of representing non-linear systems [33][3] and can be expressed as the discrete-time representation:

$$y(n) = h_0 + \sum_{i=1}^{\infty} \sum_{m_1=0}^{\infty} \cdots \sum_{m_i=0}^{\infty} h_i(m_1, \cdots, m_i) \times x(n-m_1) \cdots x(n-m_i) \tag{3.13}$$

where h_i are the so-called ith-order Volterra kernels. Volterra filters may be regarded as Taylor series approximations with memory. The advantage of the Volterra model is that the filter output is still linear in the coefficients. Thus a lot of common algorithms from linear signal processing can be applied to the Volterra filtering by way of a non-linearly extended signal vector.

Various renditions of the Volterra filter have found application in numerous fields, including echo- and noise-cancellation schemes [25][76]. Other applications include speech and signal processing [33]. In practice, the infinite summations in (3.13) above are replaced by finite ones, resulting in a non-linear filter of finite memory and order. Not surprisingly, Volterra filters of the form shown in (3.13) above are very computationally expensive.

3.2.3 Order Statistic Filters

For a random variable X with N observations denoted X_1, X_2, $X_3, \ldots, X_N$, the order statistics are obtained by sorting the independent observations $\{X_i\}$ in ascending order [74]. This sorting operation produces a new vector of weighted values $\{X_i\}$ subject to the constraint:

[3]Much of the information in this subsection was derived from this reference.

$$X_{(1)} \leq X_{(2)} \leq X_{(3)} \leq \cdots \leq X_{(N)} \tag{3.14}$$

where the values $\{X_{(i)}\}$ are the *order statistics* of the N observations. An *order statistic filter* (OSF) is an estimator

$$F(X_1, X_2, \ldots, X_N) = \alpha_1 X_{(1)} + \alpha_2 X_{(2)} + \cdots + \alpha_N X(N) \tag{3.15}$$

where α_i are the filter coefficients. For any distribution, the optimal coefficients $\{\alpha_i\}$ can be determined by minimising the criterion function

$$J(\alpha) = E[(\alpha^T X - \mu)^2] \tag{3.16}$$

where α is the vector of order statistic filter coefficients, X is the vector of order statistics, and μ is the mean. Segura et al used order statistics in [74] to obtain an estimate of the instantaneous SNR in the implementation of a speech / non-speech detection (SND[4])module in order to discriminate noise from speech. They also report significant improvements in terms of word error rate performance over competing speech enhancement techniques.

3.2.4 Histogram Equalisation

Histogram equalisation (HE) is a non-linear technique that was originally used for restoration of corrupted images [74]. In image processing, an image is divided into pixels of varying intensity (or brightness). This intensity may assume values that normally range from 0 (for black) to 255 (for white) [5]. However, the distribution of these values is often non-uniform, meaning that a significant proportion

[4]NB. the SND is sometimes called a voice activity detector (or VAD).

[5]This explanation of how HE works was provided in person by Dr Fred Nicolss of the Digital Image Processing Lab, University of Cape Town.

of pixels may have intensities within a disproportionately small range, (e.g. 5 % of pixels may have intensities in 0 to 100, 90 % between 100 to 250, and 5 % between 250 and 255). This non-uniform distribution of pixel intensities amounts to compressing the intensity scale since 90 % of all pixels would have values between 100 and 250, which virtually compresses the range to 151 (i.e. 100 to 250) from the original 256. Thus HE may be used to "decompress" the effective intensity scale by using a non-linear mapping function such that the full intensity scale (i.e. 0 to 255) is realloated to the most prevalent values. The nonlinear mapping process may be some function of the intensity histograms.

Histogram equalisation has also been successfully applied in automatic speech recognition (ASR) systems as a noise compensation technique [74][6]. In the context of ASR, it is used to compensate both linear and nonlinear distortions of the feature vector. The goal of HE is to formulate a function $x(y)$ that transforms the probability distribution of the corrupted speech signal $p_y(y)$ into a probability distribution for clean speech $p_x(x)$. Provided that $x(y)$ transforms $p_y(y)$ into $p_x(x)$, then the cumulative histograms verify that

$$C_y(y) = C_x(x(y)) \tag{3.17}$$

and thus $x(y)$ can be obtained by the cumulative histograms of the noisy speech and clean speech as:

$$x(y) = C_x^{-1}[C_y(y)] \tag{3.18}$$

where C_x^{-1} represents the inverse function of C_x.

[6]The rest of the information in this sub-section was derived from this reference.

3.2.5 Blind Deconvolution

"In blind deconvolution, a convolved version $x(t)$ of a scalar signal $s(t)$ is observed, without knowing the signal $s(t)$ or the convolution kernel" [42]. This denoising technique is called "blind deconvolution" because the convolution kernel is unknown [12]. In the context of telephony speech, $x(t)$ is the distorted speech signal, $s(t)$ the original (undistorted) clean speech signal, and $h(t)$ the convolution kernel (i.e. the channel impulse response).

In [12], blind deconvolution was used to restore a corrupted image without any explicit knowledge of the convolution kernel, but this technique reportedly had little success. The degradation process was modelled as an additive noise function plus convolution with an unknown kernel. If the additive noise component is ignored, then the degradation may be modelled as a convolution process in the time domain, which is analogous to multiplication in the spectral domain. Thus taking the log of this product yields the sum of the (unknown) convolution kernel $H(\omega)$ and the original signal $S(\omega)$ (in the spectral domain, that is). Since the degradation has been simplified to a sum of the signal and the channel impulse response, statistical estimation may now be used to estimte $H(\omega)$ and thus solve for $S(\omega)$ [12].

Unfortunately, the noise process cannot be ignored in reality. Thus the log of the product of $S(\omega)$ and $H(\omega)$ plus $N(\omega)$ ought to be estimated, where $N(\omega)$ is the noise power spectral density. The author(s) in [12] claim that the first approach that they had success with estimates $H(\omega)$ as:

$$log\,|H(\omega)| = \frac{1}{M}\sum_{k=1}^{M}[log|V_k| - log|U_k|] \tag{3.19}$$

where U_k and V_k are obtained by breaking the original image (u) and the corrupted image (v) into smaller blocks and computing their Fourier transforms (which may

be substituted with M frames of the clean speech signal $s(t)$ and the observed signal $x(t)$ for speech processing applications). However, U_k is assumed unknown, so an estimate of it may be obtained using Wiener filtering, for instance. Moreover, this technique only calculates the magnitude of $H(\omega)$, hence it is most suitable for phaseless linear shift invariant (LSI) filters.

3.3 Summary

In this chapter, the author investigated possible ways of reducing the effect of noise for speech that has been transmitted across a telephone network. Furthermore, it was found that different types of noise are present in a telephone channel. This implied that simple linear filtering techniques might not be very effective in purifying telephone speech, and hence non-linear techniques were investigated. However, some non-linear techniques, although possibly very effective, were found by the author to be somewhat undesirable since they are clearly computationally expensive and thus not suitable for real-time speaker verification applications.

Chapter 4

System Design

As explained in Chapter 1, the primary objective of this project is to design and implement a robust SV system whose performance is comparable to that of competitive architectures trained and tested under similar conditions. This will be accomplished by first implementing a baseline architecture using the concepts and methodologies outined in Chapter 2. Once the implementation has been confirmed (i.e. the results must be consistent with literature for similar systems), the baseline system will be refined using a non-linear filtering technique in order to improve the fidelity of the speech signal. This is motivated by the fact that speaker recognition systems that use clean speech (e.g. TIMIT) perform very well [19][61][31].

4.1 The Baseline System

"Given a segment of speech, Y, and a hypothesised speaker S, the task of speaker detection, also referred to as verification, is to determine if Y was spoken by S" [71][1]. Assuming that Y was generated by a single speaker, the speaker verifica-

[1] Much of the development that follows was derived from this reference.

tion problem can be formulated as a hypothesis test between

$$H_0 : Y \; is \; from \; the \; hypothesised \; speaker \; S$$

and

$$H_1 : Y \; is \; from \; a \; different \; speaker.$$

The optimum test to decide between these two hypotheses is to accept the hypothesised speaker S if the likelihood ratio $\frac{p(Y|H_0)}{p(Y|H_1)}$ is greater than or equal to some threshold θ and to reject the hypothesised speaker S if the likelihood ratio is less than this threshold, where H_0 is the hypothesis that Y was generated by speaker S, and H_1 is the hypothesis that it was not.

Figure 4.1 depicts a block diagram of a generic likelihood ratio-based speaker verification system. The input speech file is first digitised and then parameterised to generate a set of feature vectors. During the speaker registration phase, these feature vectors are used to create speaker models (i.e. the "Hypothesised Speaker Model" block) for a particular speaker. Simultaneously, an anti-speaker model is created using speech data generated by other speakers in the database (i.e. the "Background Model" block). A database of registered users is then populated using these speaker and anti-speaker models.

During recognition, the test speaker's feature vectors are analysed for proximity with a) the hypothesised speaker's model and b) the hypothesised speaker's anti-speaker model , and the difference is compared to a decision threshold. If this difference is above the threshold, the hypothesised speaker is accepted as the claimed speaker, otherwise it is rejected.

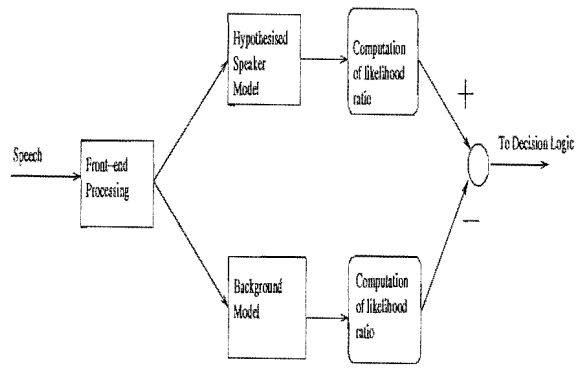


Figure 4.1: A likelihood ratio-based SV system [71]

Using this framework, the baseline architecture for this thesis project is designed as illustrated in Figure 4.2. The various blocks are explained in the subsections that follow. In the figure, DFT stands for *discret Fourier Transform*, whereas LPF stands for *Lowpass filter*[2].

4.1.1 Front-end Processing

Since the NTIMIT database will be used for all work relating to this thesis, there is no need for digitising the speech files since they are already digitised. However, each speech utterance is segmented into 20 ms frames which overlap by 10 ms [71]. Subsequently, frame energies are computed, and only frames with energies above a certain threshold are parameterised using the PFS algorithm to generate a 30-dimensional feature vector. In order to avoid ambiguity, exactly the same values of α and β will be used for both the baseline architecture and the proposed algorithm. This is so that any performance improvement obtained when the proposed algorithm is implemented can be unambiguously attributed it. Incidentally, there are various ways of suppressing noise-only frames from the speech wave-

[2]The front-end pre-processing (i.e. from "DFT" to "Cosine Transform" in Figure 4.2) was adopted from code provided by the author's supervisor, D.J. Mashao. Mashao used this code to implement speaker identification software.

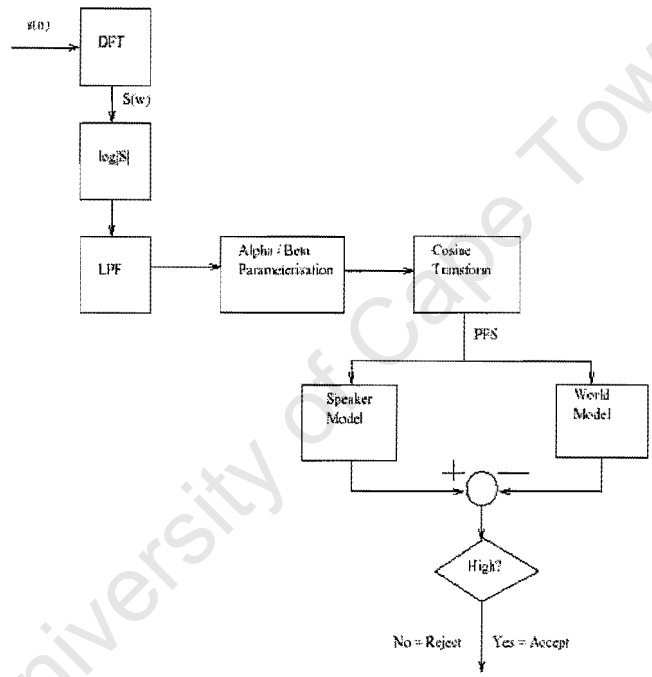


Figure 4.2: Baseline SV architecture

form. Examples include a speech / non-speech detector (SND, also called a VAD), or a simple energy-based noise flooring technique (i.e. only frames that have energies above a certain noise floor will be parameterised). The latter method will be used in this project.

However, setting the noise floor at very low values means that most noise frames will be accepted as speech frames and thus lead to poor performance, whereas choosing a high value means that most speech frames will be misclassified as noise. Consequently, more speech files will be required in order to generate the same number of feature vectors. Unfortunately, this option is not reasonable since there are a limited number of speech files in NTIMIT (i.e. ten per speaker). Thus an intermediate value is more sensible. Nonetheless, this threshold can only be determined empirically.

4.1.2 Gaussian Mixture Models

All speaker models will be created using Gaussian mixture models (described in detail in section 2.3). According to [38], the performance of a GMM-based classification engine depends on, among other things, the number of mixture models. However, many researchers report satisfactory performance when 16- or 32-mixture models are used [28][38][9]. In addition, for a set of training vectors, model parameters are estimated using the k-means clustering algorithm. This iterative algorithm improves the GMM parameters in order to increase the likelihood of the estimated model for the observed feature vectors. That is, for consecutive iterations k and $k+1$, the likelihood ratio $p(X|\lambda_{k+1})$ is marginally greater than the likelihood ratio $p(X|\lambda_k)$. However, subsequent iterations of the k-means algorithm improve the likelihood by a diminishing amount, suggesting that the likelihood converges to a common value. In practice, about ten iterations are adequate to guarantee convergence.

4.1.3 World Models

According to [71], there is no objective measure to determine the right population of speakers or amount of speech to use in training a universal background model (or UBM, which is similar to a world model). In addition, it is also reported in this reference that there was no performance loss when the amount of speech used to train a UBM was reduced from six hours to one hour.

On the other hand, the number of Gaussians for a world model has a direct bearing on overall recognition performance [49]. In [49], Bengio et al propose a simple technique to determine the optimum number of Gaussians for a world model. In essence, they trained a world model using 90 % of the available training speech data and continuously varied the number of Gaussians. Subsequently, they selected the world model that yielded the highest likelihood using the remaining training speech data (i.e. 10 %). In other words, they created world models with, say, 32, 256, 512, 2048, etc, Gaussians and tested them using the remaining speech data.

For instance, suppose that one wishes to determine the ideal number of Gaussians in speaker FAKS0's world model. Since the objective is to maximise the expression $log\ p(X|\lambda_{FAKS0}) - log\ p(X|\overline{\lambda}_{FAKS0})$ where $\overline{\lambda}_{FAKS0}$ represents the speaker's world model, the number of Gaussians that minimise the second term will be selected as the optimal one for the world model. Thus the optimal number of Gaussians in a world model is best determined empirically.

4.1.4 Decision Logic

As stated in section 1.4.3, there is too little data in NTIMIT for some of it to be reserved for threshold calibration purposes. This implies that a global threshold will be determined, and all scores will be compared against it for authentication.

One of the methods of determining this threshold entails calculating the minimum and maximum client scores during testing and adjusting it in small increments across the entire range of scores. In this project, these scores will be calculated using the Mahalanobis distance measure for both speaker and world models. At each iteration, the values of FAR and FRR are noted. The process is repeated until the absolute difference between FAR and FRR is zero (or minimal, in which case the EER is estimated using an technique known as the *half-total error criterion*, or HTER). The HTER estimates the EER by simply averaging the FAR and FRR values when the difference between them is minimal. A script that calculates FAR and FRR values is provided in Appendix A.

Section 4.2 presents the theory behind the non-linear polynomial approximation (PA) filter that the author proposes in order to denoise telephone speech. Although there are many non-linear filtering techniques (e.g those presented in Chapter 3), the proposed filter was chosen because of its simplicity and versatility.

4.2 Least-squares Polynomial Approximation

As stated in section 3.2, non-linear filters are generally more effective than linear ones in denoising signals that have been corrupted with heterogeneous noise processes. However, the non-linear filtering techniques reviewed in the previous chapter have either already been implempted in speaker recognition (e.g. non-linear spectral subtraction) or they were considered to be too computationally expensive (e.g. Volterra filters) to be used in a real-time authentication scenario such as speaker verification. Thus the author decided to implement a simple, non-linear technique known as the polynomial approximation (PA) algorithm. The theory of this algorithm will be presented shortly.

It is sometimes desirable to fit an analytical function to experimental data in order to compensate for errors. This procedure is known as data fitting and uses a func-

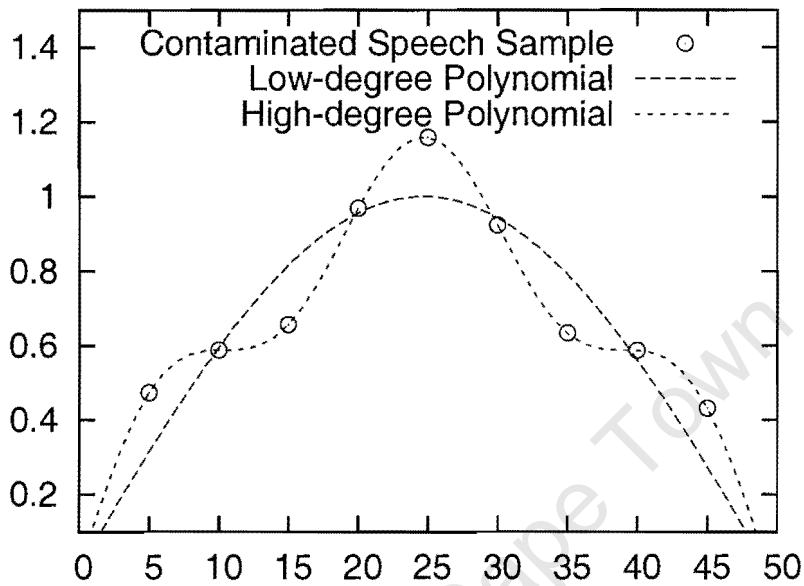


Figure 4.3: Illustration of Smoothing Effect of A Lower-Order Polynomial

tion such as an interpolating polynomial to fit the data. However, it is obviously undesirable to replicate data that is known to be noisy since that is equivalent to reproducing the noise. In this case, a polynomial that approximates the data under constrained conditions (i.e. least squared error) might be more useful. This is illustrated in Figure 4.3.

In Figure 4.3, a synthetic speech waveform is described using two interpolating polynomials of different orders. The high-order polynomial describes the data exactly (i.e. it passes through all the data points), whereas the low-order polynomial filters the data by only passing through some points.

According to interpolation theory, any arbitrary set of $m+1$ data points can be described using a unique mth degree polynomial. If the data points are known to be

error-free, then an mth degree polynomial would suffice[26][3]. However, in most practical situations the data will most likely be contaminated by noise, suggesting the use of a lower-order polynomial. Assuming $m+1$ sampling instants denoted $x_0, x_1, \ldots, x_m$ corresponding to samples $y_0, y_1, \ldots, y_m$, then an nth-order (n < m) polynomial $f(x)$ might be made to fit the data as follows:

$$f(x) = \sum_{j=0}^{n} a_j x^j \tag{4.1}$$

Hence the error (i.e. difference between the observed data and the polynomial) at point x_i is:

$$e_i = y_i - f(x_i) \vee i = 0, 1, 2, \ldots, m \tag{4.2}$$

Thus the goal is to find the filter coefficients a_j in (4.1) that minimise the error at each data point x_i. But it is not possible to make the error zero at every point because a lower-order polynomial cannot go through all the points as depicted in Figure 4.3. However, a least-squares criterion may be used to *minimise* the sum of the squared error at each data point. That is, the least-squares criterion minimises

$$E = \sum_{i=0}^{m} e_i^2 = \sum_{i=0}^{m} [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n)]^2 \tag{4.3}$$

But minimisation requires that the following conditions be fulfilled:

$$\frac{\partial E}{\partial a_0} = 0, \frac{\partial E}{\partial a_1} = 0, \ldots, \frac{\partial E}{\partial a_n} = 0. \tag{4.4}$$

This is a set of $n+1$ equations and as many unknowns (i.e. the filter coefficients

[3]The development that follows was derived entirely from this reference.

$a_0, a_1, \ldots, a_n$), and thus the coefficients can be determined analytically using a suitable technique. In this thesis, these equations will be solved using matrix inversion since this algorithm was already available. Differenting (4.3) with respect to the $n+1$ filter coefficients $a_0, a_1, \ldots, a_n$ above yields:

$$\begin{aligned}
\frac{\partial E}{\partial a_0} &= 0 = \textstyle\sum_{i=0}^{m} 2[y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n)](-1) \\
\frac{\partial E}{\partial a_0} &= 0 = \textstyle\sum_{i=0}^{m} 2[y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n)](-x_i) \\
&\vdots \\
\frac{\partial E}{\partial a_0} &= 0 = \textstyle\sum_{i=0}^{m} 2[y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n)](-x_i^n).
\end{aligned} \tag{4.5}$$

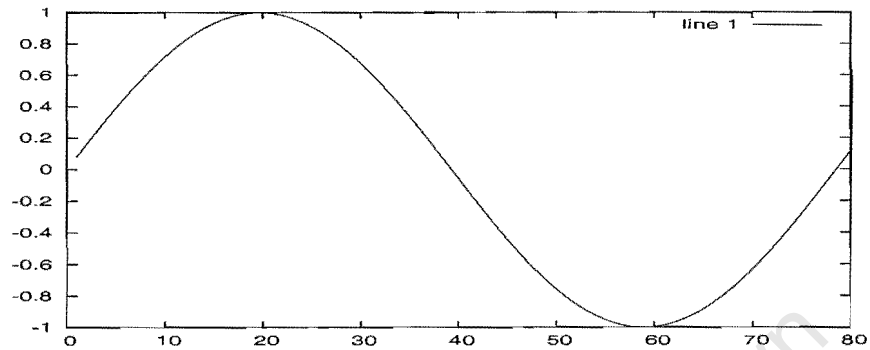
The above set of equations is equivalent to

$$\begin{aligned}
C_{00}a_0 + C_{01}a_1 + C_{02}a_2 + \cdots + C_{0n}a_n &= \textstyle\sum_{i=0}^{m} y_i, \\
C_{10}a_0 + C_{11}a_1 + C_{12}a_2 + \cdots + C_{n}a_n &= \textstyle\sum_{i=0}^{m} x_i y_i, \\
&\vdots \\
C_{n0}a_0 + C_{n1}a_1 + C_{n2}a_2 + \cdots + C_{nn}a_n &= \textstyle\sum_{i=0}^{m} x_i^n y_i,
\end{aligned} \tag{4.6}$$

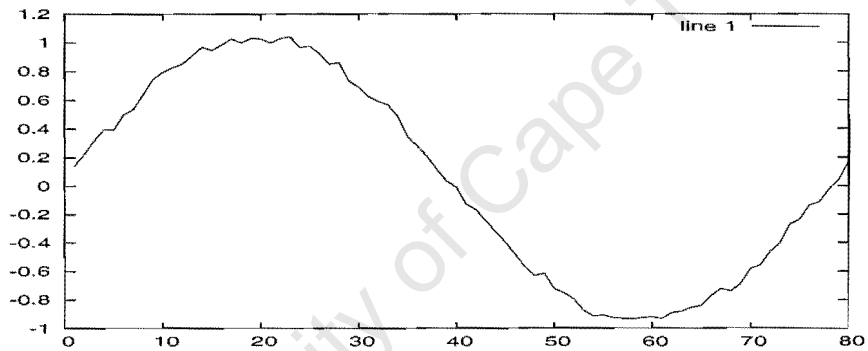
where

$$C_{ij} = \textstyle\sum_{k=0}^{m} x_k^{i+j}. \tag{4.7}$$

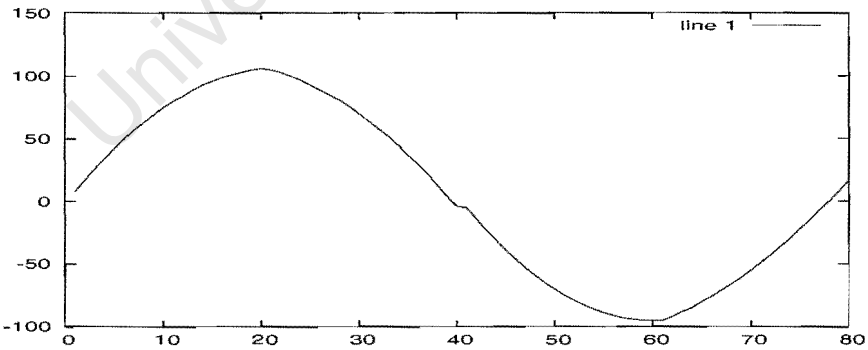
Figures 4.4 and 4.5 demonstrate that this algorithm is able to remove both additive noise and impulsive noise. Figure 4.4 (a) depicts an unperturbed sinusoidal waveform, while Figure 4.4 (b) depicts an additive random noise-contaminated version of it. This contaminated signal was then denoised with the above-mentioned algorithm using an analysis window of 20 samples and a quadratic PA filter to generate the denoised version depicted in Figure 4.4 (c). It is clear that the proposed



(a) Original sine wave signal



(b) Additive random noise-contaminated sine wave signal



(c) Denoised version of corrupted signal (not normalised)

Figure 4.4: Removal of additive random noise using the PA filter

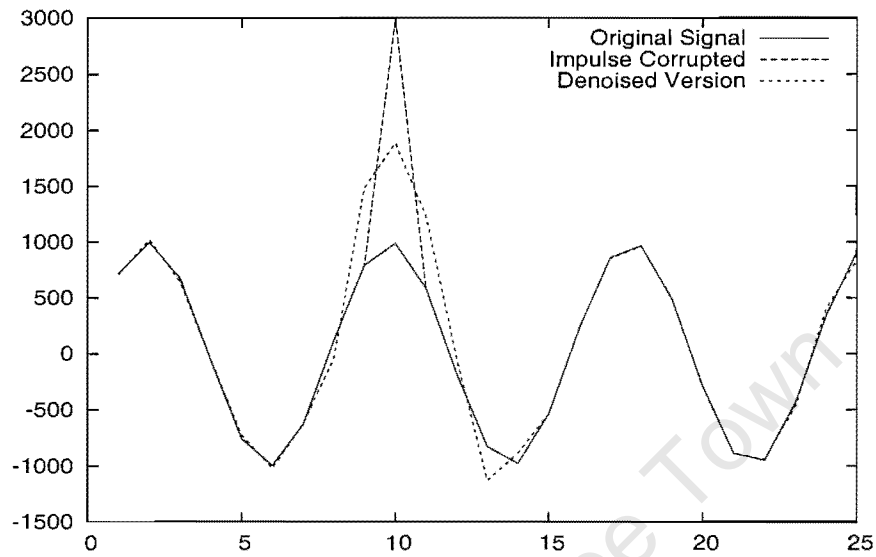


Figure 4.5: Illustration of impulse noise decontamination using the proposed filter

filter is able to remove additive noise quite effortlessly (save for a slight phase distortion at t = 40).

On the other hand, Figure 4.5 shows a slow-varying sinusoidal waveform that was artificially corrupted using a high-amplitude impulse at t = 10 to generate an impulse-corrupted version [57]. The corrupted signal was decontaminated using an analysis window of 7 samples and a 4^{th} order PA filter without the use of an impulse detection module in contrast to Wang and Zhou's method in [79]. Quite clearly, only samples in the immediate vicinity of the impulse were affected by the filtering operation. Other samples were virtually unaffected by the filtering process.

The subsequent section provides a review of signal processing applications in which similar filters have been (or could be) applied successfully.

x:	0.0	0.5	1.0	1.5	2.0
y:	-2.9	-1.8	0.2	2.0	5.1

Table 4.1: Uncompressed Data

4.3 Other Applications

Least-squares based polynomial approximation (PA) has an extensive range of signal processing applications, some of which include data compression. In section 4.3.1, the author illustrates how this technique may be used to achieve data compression.

4.3.1 Data Compression Using Polynomial Approximation

Suppose that one would like to "compress" the data in Table 4.1[4].

Assuming a quadratic interpolating polynomial, equation (4.6) yields the following system of equations:

$$5.0a_0 + 5.0a_1 + 7.5a_2 = 2.7$$
$$5.0a_0 + 7.5a_1 + 12.5a_2 = 12.7$$
$$7.5a_0 + 12.5a_1 + 22.125a_2 = 25.05$$

which can be solved using matrix inversion (or any other relevant method) to yield:

$$a_0 = -2.889,\ a_1 = 1.714,\ a_2 = 1.143$$

[4]These values were derived from [26], but not the actual example showing how data compression may be achieved.

x:	0.0	0.5	1.0	1.5	2.0
P(x):	-2.9	-1.7	0.0	2.3	5.1

Table 4.2: Illustration of PA-based data compression

so that the quadratic polynomial becomes

$$P(x) = -2.889 + 1.714x + 1.143x^2. \tag{4.8}$$

Using (4.8), the results shown in Table 4.2 are obtained. It is obvious that the PA approach introduces some fractional error according to Table 4.2. However, if perfect fidelity is not absolutely essential, this approach offers an interesting alternative to conventional data compression schemes. In this case, one would only need to store the coefficients a_0, a_1 and a_2 instead of the five data points in Table 4.1. Unfortunately, better fidelity can only be obtained at the expense of lower compression rates (i.e. higher polynomial orders).

4.3.2 Image Restoration

In [79], a polynomial approximation (PA) filter is used to restore images that have been corrupted by impulse noise. This is accomplished by applying an impulse detection algorithm to the image data on a pixel-by-pixel basis. The impulse noise might be generated by imperfect sensors or a noisy communication channel [79]. However, care must be taken when this type of filter is used because if it is applied on clean data it could destroy important information (e.g. edges could be blurred).

4.3.3 Detail Concealment

In addition to noise suppression, polynomial approximation or smoothing may also be used to make data *look* visually more appealling [53], such as in Figures

4.6 (a) and 4.6 (b) [51]. This demonstrates that the technique may also be applied on uncontaminated data. However, it is easy to see how the technique may be used to smooth noisy data by observing the two figures.

A comparison of Figures 4.6 (a) and 4.6 (b) reveals that the latter looks more appealing because unwanted detail has been concealed. However, it is also clear that the smoothing algorithm may obscure wanted detail if applied without care. Perhaps a better solution would be one in which the degree of smoothing is determined adaptively depending on local noise content. In other words, regions that are heavily contaminated would require "hard smoothing", whereas high-SNR regions might not require any smoothing.

4.4 The Proposed PA Filter-Based Architecture

The proposed PA-based noise suppression algorithm is depicted in 4.7. In this architecture, a contaminated speech file is pre-processed on a subframe basis using an adaptive algorithm to determine the appropriate filter order. The subframe length (SFL)[5] is chosen so as to minimise computational overhead. For instance, an SFL of 320 (corresponding to the number of samples in a typical 20 ms speech frame and a sampling rate of 16 kHz) would introduce excessive computational overhead (e.g. inverting a 320 x 320 matrix), whereas a smaller SFL offers faster computation but at the expense of more subframes.

Having determined the SFL, the energy in each frame is computed and compared to a universal threshold, which ought to be determined beforehand. If the energy in a particular frame exceeds the threshold, then it is assumed that the local SNR is high, and thus no filtering is required. This is equivalent to setting the filter

[5]The *subframe length* is the number of samples in an analysis window.

(a) A low-detail RBF-smoothed image

(b) A high-detail RBF-smoothed image

Figure 4.6: Illustration of smoothing for image processing

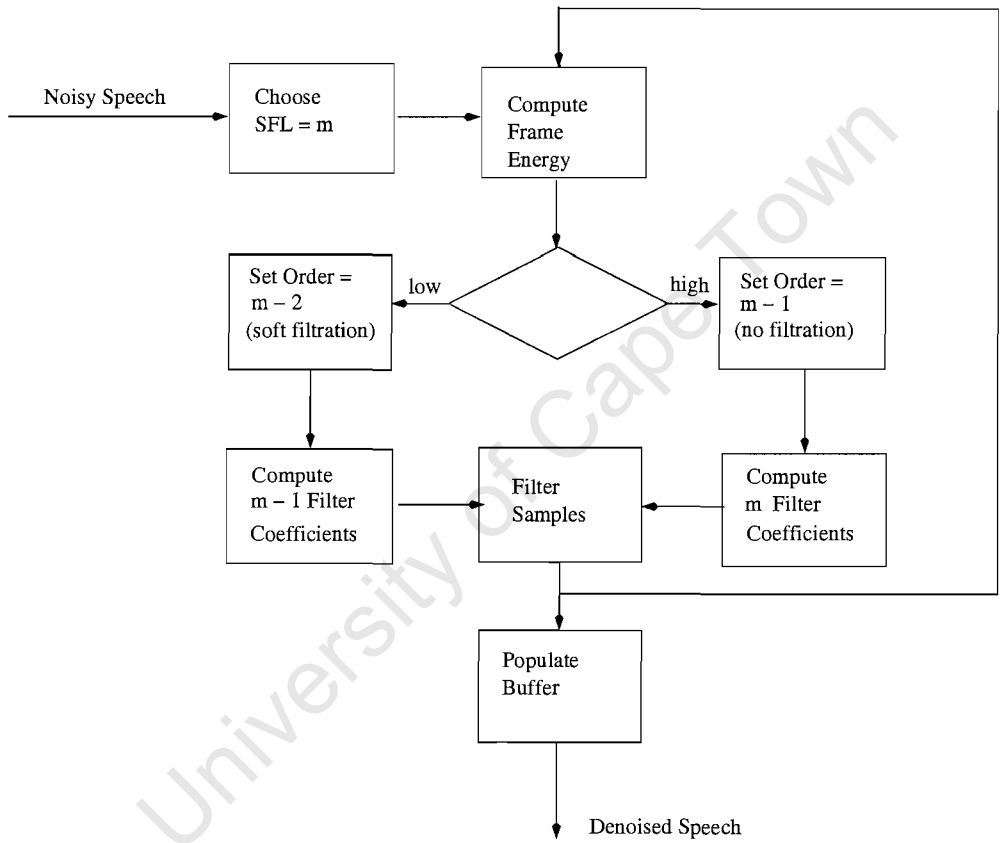


Figure 4.7: A block diagram of the proposed architecture

order at $m = SFL - 1$ (where m is the filter order) in accordance with the interpolation theory outined in section 4.1. Otherwise, if the frame energy is low, the filter order is set to $m = SFL - 2$. In principle, lower filter orders could be used if the frame energy is very low, but they generally tend to destroy essential speaker specific information, and the overall performance suffers as a result. That is why a relatively high filter order of $m = SFL - 2$ is recommended even for low-energy frames. It must be pointed that all the subframes in the same frame are processed with the same filter order since the energy is computed for each frame and not for each subframe.

After determining the appropriate filter order, the filter coefficients are computed, and the subframe samples are then filtered. The filter output is stored in a buffer of denoised samples, and the process is repeated until the entire speech file has been processed. The denoised speech samples are then processed as illustrated in Figure 4.2 to generate PFS features.

4.5 Design Constraints and Criteria

In summary, the proposed design was chosen in accordance with the criteria layed out in the following sub-sections.

4.5.1 Real-time Performance Capability

The proposed design must not compromise real-time processing capability. **In other words, an algorithm that promises great performance at the expense of computational feasibility would be considered unattractive for this work.** The reason for this is that, in a fielded telephone-based SV system, clients would be reluctant to make use of the service if the authentication process was too long,

especially if they also had to pay for the call (i.e. assuming that there was no 0800 service available).

4.5.2 Reproducibility

It is imperative that the proposed algorithm be implemented in a transparent and reproducible fashion. **This means that it should not be optimised for a particular database since different conditions (or databases) might elicit poorer performance**. Otherwise it would be difficult to prove that any performance improvement resulting from an application of the algorithm is not incidental. Stated differently, the algorithm should process data transparently irrespective of where it originates. Although this methodology might not be preferable in view of the fact that unique problems sometimes require unique solutions, it does however guarantee that any improvement obtained by applying the alogrithm on specific data can be extrapolated with some confidence to data that has yet to be tested.

4.5.3 Robustness

As stated at the beginning of Chapter 1, it is highly desirable for speaker recognition architectures to perform consistently well regardless of where the recognition task might be required. Although state-of-the-art SV architectures perform very well in ideal (e.g. labaratory type) environments, it is much more difficult to even approximate this performance in real-world scenarios, such as in a soccer stadium during a football match, or over a telephone network. Thus the proposed solution must perform verifiably better in less-than-ideal conditions, specifically over a telephone network.

4.6 Summary

This chapter presented architectural descriptions of both the baseline system and the proposed PA filter-based algorithm. The baseline design was based on a framework proposed by Reynolds for likelihood ratio-based SV architectures. Furthermore, potential applications of the PA algorithm are also cited and discussed (e.g. data compression). Finally, system design criteria are specified. These criteria serve as general guidelines for the author since it is reasonable to expect that some modifications might be required, especially during the implementation phase. These modifications must therefore be effected in accordance with the stated guidelines.

Chapter 5

Experimental Work and Discussion of Results

As outlined in Chapter 2, the performance of an SV system may be calibrated in terms of several performance metrics, of which the EER is most popular. While it is true that the EER is not a realistic evaluation metric since its computation is performed *a posteriori* using actual (as opposed to theoretical) FAR and FRR values [30], it does however provide a reasonable basis for the comparison of different SV architectures because of its widespread use. In any case, the results of this project are also expressed in terms of a different evaluation metric (the D') in order to ascertain their validity. **In other words, if the EER and the D' yield conflicting results, the proposed algorithm's superiority would be ambiguous. Otherwise its superiority would be confirmed provided both metrics indicate an improvement in performance.**

As described in Chapter 2, the D' (pronounced 'dee prime') metric provides an indication of the separability of two distributions (see Figure 5.1). In the context of speaker verification, the overlapping distributions in Figure 5.1 represent client and impostor scores. The distributions at the top of the figure correspond to

contaminated data (i.e. noise causes impostors to be mistakenly accepted as true clients and vice versa). This is explained by the large overlap between the two distributions (i.e. low D' values). In general, there is a correlation between the extent of overlap and recognition error. In fact, as the signal strength increases, the D' also increases [5]. Therefore an SV system that uses clean (i.e. high-SNR) speech (e.g. TIMIT) could have distributions like those shown at the bottom of the figure (i.e. high D' values).

It is therefore reasonable to expect lower recognition errors to be associated with distributions that are more separable (i.e. higher D' values), while higher error rates are associated with not-so-separable impostor and client score distributions. Thus, in the context of this project, the D' is used to confirm the performance improvement (expressed by way of the EER) obtained by refining the baseline architecture as described in Chapter 4.

5.1 Simulation Conditions and Parameter Settings

All simulations relating to this thesis were performed on a 1.7 GHz AMD Linux machine using the C programming language and Python scripts. In addition, the entire test corpus part of NTIMIT which consists of 112 male and 56 female speakers was used for all simulations. This means that there were a total of 168 speakers from all 8 dialect regions (i.e. DR1 to DR8). Each speaker's model was created using about 21 seconds of training speech (i.e. 7 sentences, each lasting roughly 3 seconds) and 9 seconds of testing speech (i.e. 3 concatenated sentences, each also about 3 seconds long). Moreover, a world model was generated for each speaker using about 2505 seconds of speech (i.e. 167 speakers in a generic world model * 5 sentences for every speaker in the world model * 3 seconds per sentence).

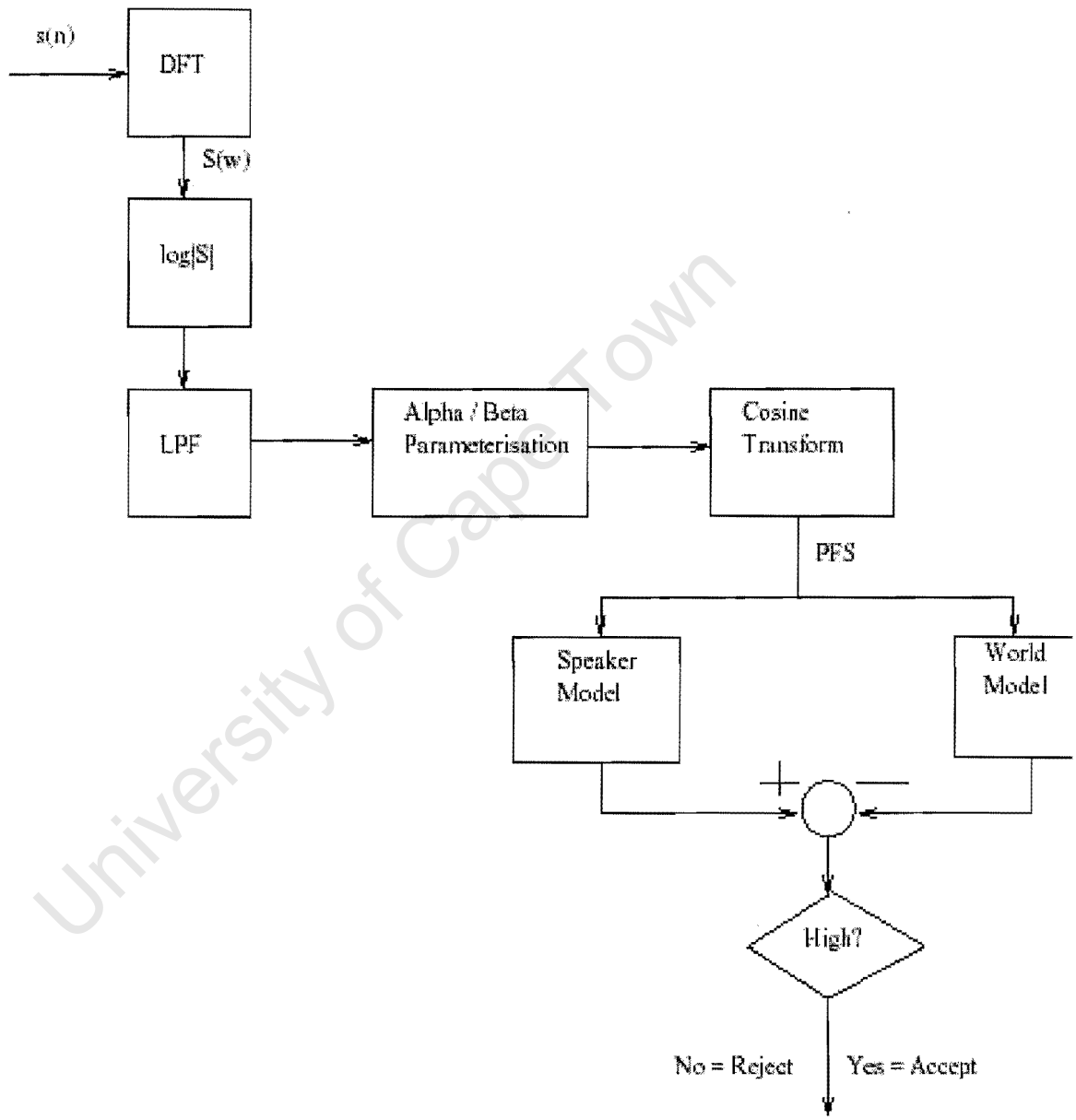


Figure 5.1: Illustration of the D' metric [5]

Furthermore, each sentence was broken down into 20 ms frames, which translates to 320 samples per frame (corresponding to a sampling rate of 16 kHz). Each frame was then parameterised using PFS to obtain a 30-dimensional feature vector. The optimal values for α and β were found to be 4 and 1.6, respectively [58]. Thereafter, 32-mixture Gaussian mixture models were created for all speaker models using 25 iterations of the k-means algorithm.

5.1.1 Determining the Number of Mixtures in a World Model

In accordance with the method described by Bengio and Le in [49], the optimal number of mixtures in a generic world model was determined empirically by computing the likelihoods corresponding to different numbers of mixtures. In this project, the number of mixtures was chosen as 32, 64, 128, 256 and 512 using speaker FAKS0's true identity claims (i.e. as opposed to impostor claims). The results of this test are depicted in Figure 5.2. Incidentally, the number of mixtures for the corresponding speaker model was fixed at 32. As stated in chapter 4, the objective of this test was to find the number of mixtures that maximised the expression $log\ p(X|\lambda_{FAKS0}) - log\ p(X|\overline{\lambda}_{FAKS0})$, where $\overline{\lambda}_{FAKS0}$ represents the speaker's world model.

In addition, the author only used a single speaker (i.e. FAKS0) to determine the optimal amount of mixtures since it was logical to assume that these observations were representative of the expected behaviour for all other speakers. In other words, it is quite likely that the graph in Figure 5.2 would also be obtained for any other speaker. Furthermore, the amount of time required by the k-means clustering algorithm to generate these mixtures was almost prohibitive for large numbers of mixtures (see Table 5.1). Thus 32 mixtures were also used for the world models.

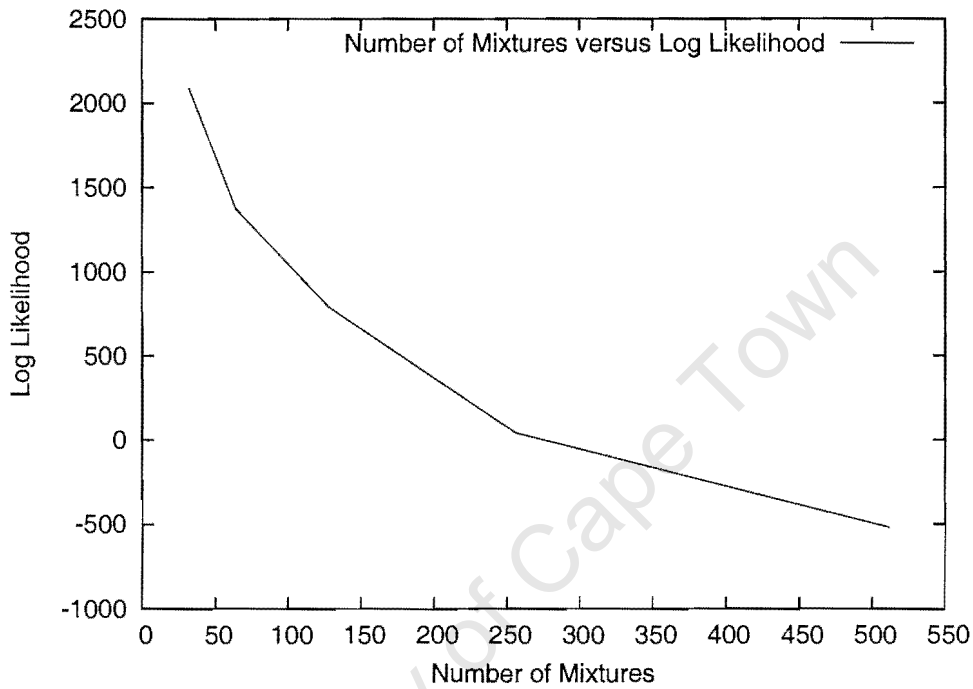


Figure 5.2: Determining the optimal number of mixtures in a world model

Mixtures	Duration [s]
32	242
64	331
128	527
256	993
512	1,811

Table 5.1: Training times for different numbers of mixture models

5.1.2 Selection of Files

The NTIMIT database used in this project has a total of 630 male and female speakers, each of which has 10 utterances. These utterances include 2 dialect calibration files (i.e., SA files), which are identical for everyone in the database. In addition, each user also has 5 phonetically compact (SX) and 3 phonetically diverse (SI) files. However, some SX files are the same for certain speakers, while all SI files are unique [53]. Since this project seeks to improve the performance of a text-independent SV system, the author decided to use SA and SX files for training while SI files were used for testing as in [55]. This means that altogether 7 files were used as training material, whereas the remaining 3 were used for testing. Owing to the fact that text-independent speaker recognition requires more data than text-depedent recognition [45], it was considered best to use all 3 files simultaneously during testing (i.e. approximately 9 seconds of concatenated test speech per speaker) as opposed to 3 separate verification transactions. By contrast, 9 utterances were used for each verification transaction in [31]. This is because there is a direct correlation between test utterance duration and recognition performance [59].

5.2 Algorithm Verification

After implementing the proposed PA filter-based algorithm, a portion of an NTIMIT file was processed in order to verify that the algorithm was working as expected. As seen in Figure 5.3, the denoised speech waveform looks much like the original NTIMIT signal since a 5^{th}- order (i.e. high-fidelity) filter was used.

By contrast, the baseline architecture was verified simply by comparing the corresponding EER result with others reported in literature (see Table 5.2). Initially, only a single run of the baseline algorithm was used in order to confirm that the implementation had been executed properly. Interestingly, the author(s) in [2] obtained an EER of 4.8 %, which is identical to the EER obtained by the author if a

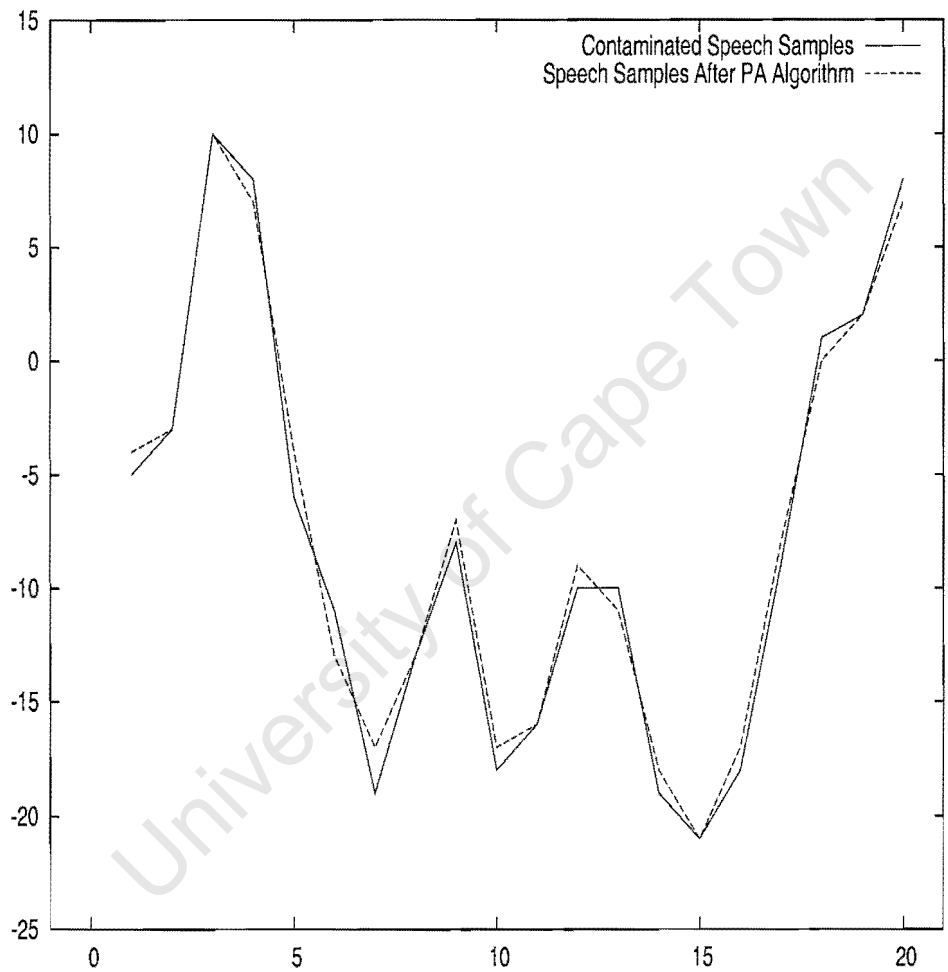


Figure 5.3: Illustration of Polynomial Approximation Algorithm on Noisy Speech

Author	Architecture	Population	EER [%]
Limpho Mothae	GMM / PFS	168	4.76
(unknown)[2]	unknown	350	4.8
Misra et al [55]	ANN / LPCC	38	6.6
Drygajlo et al [27]	unknown	400	9.9

Table 5.2: Comparison of different EER values for NTIMIT

single decimal place is used (i.e. 4.76 % rounds off to 4.8 %).

5.3 Impostor Attack Scenarios

There are several methods of calculating the FAR of an SV system. This is generally accomplished by simulating a) random impostor attacks, b) massive attacks whereby many impostors target a specific victim, or c) a few impostors attack a client [?]. The massive impostor attacks methodology whereby each user is a potential impostor to every other user undoubtedly represents a worst-case scenario and was thus adopted for this project. This is in direct contrast to the cohort approach in which it is assumed that successful attacks are likely to originate from similar-sounding speakers.

Since only the test corpus part of NTIMIT (which consists of 168 male and female speakers) was used in this project, there were a total of 168 FRR bids and 28056 (168 * 167) FAR bids. By contrast, Reynolds in [67] used 334 FRR bids and 52538 FAR bids.

5.4 Universal Decision Thresholds

Ideally, an independent decision threshold should be used for each speaker in order to minimise the EER [18]. However, this requires a dedicated calibration set

of utterances in addition to training and testing sets in order to compute speaker-dependent thresholds. Unfortunately, there is too little data in NTIMIT for some of it to be reserved exclusively for calibration purposes [52]. As a result, all verification transactions were calibrated against a universal threshold which was determined empirically. Initially, this was accomplished by plotting distributions of impostor and client scores and determining the point of intersection. These distributions were estimated from separate histograms of impostor scores and client scores. In other words, the distributions of client and impostor scores were superimposed on the same axis so as to reveal the point of intersection. However, this method was soon abandoned because it was too time-consuming. Therefore a new method was proposed in which an initial threshold value was "guessed". The proportion of client rejections and impostor acceptances above and below this threshold were noted, and thus the EER threshold was obtained by incrementing (or decrementing) this threshold. This method was much faster than the one stated earlier and was carried out using the script in Appendix A.

5.5 EER Results

Table 5.2 shows how the baseline system compares with other architectures reported in literature (all using NTIMIT). The SV system designed by Misra et al was implemented using artificial neural networks (ANN) and tested using only 38 speakers in dialect region 1 (DR1) of the NTIMIT "train" corpus. The front-end that was used consisted of LPC-based cepstra. In any event, it is doubtful whether the small population size would provide enough speaker variability for the claimed EER of 6.6 % to be sufficiently indicative of the architecture's real performance.

However, while it is true that population size should not really matter since SV is a binary decision problem (as opposed to SI, which is a $1/N$ decision

problem), it is also true that a small population size might give overly optimisitic (or pessimistic) results owing to a non-representative distribution of speakers. For instance, since the 38 speakers were all from the same dialect region (i.e. DR1), it is possible that the inclusion of speakers from other dialect regions might lower the EER somewhat since speakers from the same dialect region should sound more or less the same comparatively. This is somewhat analogous to using an SV system on a population that consists entirely of cohorts.

On the other hand, El-Maliki and Drygajlo used 400 speakers from the NTIMIT database, but they also added an artificial noise source at varying SNRs to the speech data. However, they provide no explicit information about the features that they used. The anti-speaker modelling methodology that they used (e.g. cohort or UBM approach) has also not been specified, and the amount of training speech (both for speaker and anti-speaker models) has also not been mentioned. Nonetheless, they mentioned that two sentences were used during the testing phase, but one cannot infer from this that the remaining files were all used for training since some could have been used to calibrate speaker thresholds.

In [Korean]*****, the author(s) claimed an EER of 4.8 % for NTIMIT, but there was no explicit mention of the features used, nor was the classification engine specified. However, it is reported that a population size of 350 speakers was used. Incidentally, this information was provided in Korean and translated by the author using an on-line translation utility. The uniform resource locator (url) for the translation utility is http://www.worldlingo.com/wl/Translate. To perform the translation, the url of the page that must be translated has to be provided, and the target language (i.e. English) must also be specified.

Figures 5.4 and 5.5 depict the *receiver operating characteristics* (ROC) curves corresponding to the baseline architecture and PA algorithm, respectively. These curves were obtained by adjusting the decision threshold from 1000 to -1000 in

steps of 250. This large increment was chosen so as to accelerate convergence towards the EER. When the difference between the two error rates was small, this increment was reduced to 10 in order to identify the exact point of convergence (i.e. where FAR = FRR). This variation of the increment accounts for the "kinks" around the (5 %, 5 %) ordinates in the plots since a smaller increment reveals more detail than a larger one. In addition, adjusting the decision threshold by minute increments means that sometimes the FRR does not change because of a sparse distribution of true client scores (compared to the distribution of impostor scores). This is reflected in the ROC curves as small horizontal sections.

By the way, the ROC curves corresponding to the PA algorithm appear to be a little more "bowed out" than those corresponding to the baseline algorithm as expected, although in some cases this difference is difficult to observe. This observation is consistent with the fact that a higher signal strength will cause the curves to "bow out" (i.e. make them closer to both axes) [5].

However, it is not always possible to find FAR and FRR rates that match exactly when calculating the EER [23]. **This is because the distribution of scores for both true clients and impostors is never continuous since the number of speakers in any database will always be finite**. Moreover, the number of FAR and FRR tests is another important consideration. For instance, in this project there were a total of 168 concatenated test files (corresponding to as many speakers), and consequently 168 FRR transactions. This means that the smallest amount by which the FRR can be adjusted is 1/168, or 0.595 %. Moreover, if the number of FAR (i.e. impostor acceptance) tests is considerably larger, the FAR can be adjusted by much smaller increments.

As explained in section 5.3, a massive impostor attack methodology in which each user is a potential impostor to every other user was adopted for this project since it represents a worst-case scenario. Thus there were a total of 28056 (i.e. 168 * 167)

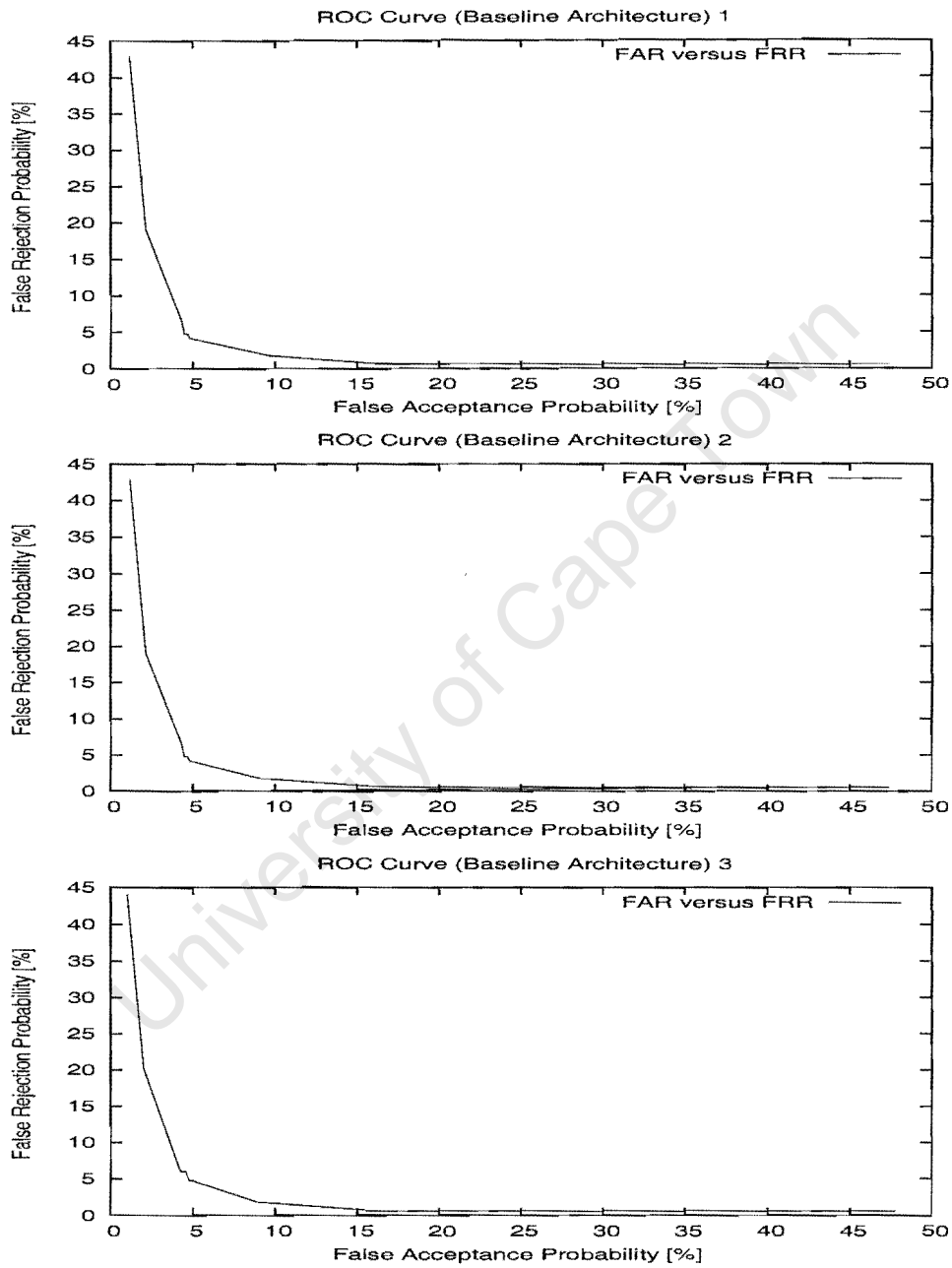


Figure 5.4: ROC curves corresponding to the baseline architecture

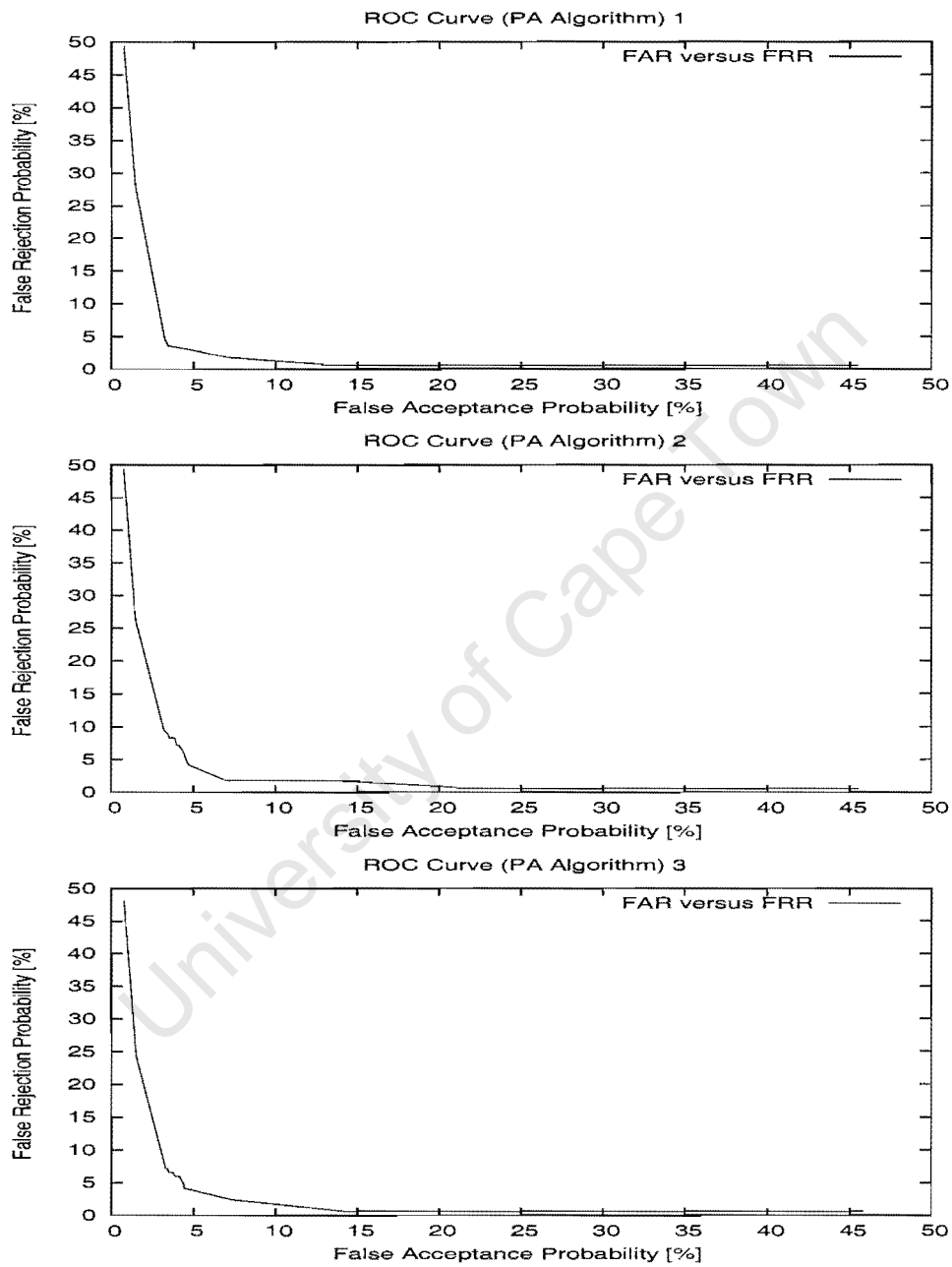


Figure 5.5: ROC curves corresponding to the PA algorithm

(a) EER values for the baseline architecture

Run	EER [%]
1st	4.76
2nd	4.71
3rd	4.21
Average	**4.56**

(b) EER values for the proposed algorithm

Run	EER [%]
1st	4.17
2nd	3.57
3rd	4.68
Average	**4.14**

Table 5.3: EER averages for the baseline and PA architectures

FAR transactions. As a result, the FAR can be adjusted by as little as $1/28056$, or 0.0036 %. Thus the FRR might overshoot (or undershoot) the FAR. In such situations, the HTER (half total error rate) is normally used to estimate the EER provided the FAR and FRR are as close as possible [23].

Tables 5.3 (a) and (b) show the average EER values obtained using the baseline architecture and the PA algorithm, respectively. Incidentally, both architectures were implemented using equal amounts of training and testing speech data, as well as equal amounts of anti-speaker modelling (i.e. world model) speech. This was done so that the resultant improvement in performance could be attributed exclusively to the superiority of the proposed algorithm.

5.6 D' Results

Table 5.4 (a) shows the D' values corresponding to three client / impostor distributions obtained using the baseline architecture, while Table 5.4 (b) shows the D' values for three client / impostor distributions corresponding to the proposed algorithm. **It is obvious that the D' figures corresponding to the PA filter-based algorithm are consistently higher than those obtained using the baseline architecture**. Although the difference between the two D' averages is only fractional (i.e. 1.78 minus 1.75), it does mean that, on the whole, distributions obtained using the PA filter are slightly more separable than those obtained using the baseline architecture. **This confirms that the reduction in EER was not incidental**.

In Table 5.5, "Impostors A" refers to the first-run distribution of impostor scores, "Impostors B" refers to the second-run distribution, etc. Interestingly, impostor scores obtained using the baseline architecture are, on average, slighltly lower than those obtained using the proposed algorithm, whereas the converse is true of client scores.

Table 5.5 shows the average times required to a) generate a speaker model and the corresponding world model, and b) perform a typical verification transaction using three concatenated SI files for both the baseline architecture (abbreviated BLA) and the polynomial approximation filter-based method (abbreviated PA). Not surprisingly, the PA algorithm is more computation-intensive than the baseline version as can be seen in Table 5.5. Incidentally, the operation that took the longest time during the model generation phase was the k-means algorithm since it only computes models pertaining to a particular set of feature vectors, regardless of how the features were generated. The rest of the time was obviously spent creating the feature vectors themselves. In other words, the BLA required about 120 seconds to generate the features, whereas the PA algorithm took about 600

(a) D' values corresponding to the baseline architecture

Distribution	Mean	Standard Deviation	D'
Impostors A	-2774.0978	2304.6309	
Clients A	208.2249	782.1239	
			1.73
Impostors B	-2785.6896	2283.3719	
Clients B	180.5952	747.850	
			1.75
Impostors C	-2774.2977	2293.4116	
Clients D	266.2024	849.2805	
			1.76
Average			**1.75**

(b) D' values corresponding to the PA algorithm

Distribution	Mean	Standard Deviation	D'
Impostors A	-2828.248	2258.1245	
Clients A	171.351	780.7643	
			1.78
Impostors B	169.503	816.7187	
Clients B	-2832.4995	2246.5875	
			1.78
Impostors C	-2833.0737	2254.6835	
Clients D	136.3512	738.001	
			1.77
Average			**1.78**

Table 5.4: D' values for the baseline and PA architectures

Architecture	Operation	Duration [s]
BLA	Train	240
PA	Train	727
BLA	Test	0.4
PA	Test	2

Table 5.5: How the PA algorithm compares with the baseline computationally

seconds. The remaining 120 odd seconds were taken by the k-means algorithm (corresponding to 32-mixture world models), which processes feature vectors regarded of how they were generated.

Furthermore, while it is true that the PA filter-based algorithm returned better results in terms of overall recognition performance, it is also true that a few clients who were properly verified by the baseline architecture were rejected by the PA algorithm. Conversely, a few impostors who were previously rejected by the baseline architecture were accepted when the PA algorithm was used. Naturally, all models that were created using a particular architecture were also tested using that same architecture (i.e. matched conditions). However, the author applied the PA algorithm on a handful of models that were created using the baseline architecture (i.e. mismatched conditions), and in a few cases verification errors were corrected. These observations prompted the author to propose a new decision-fusion type algorithm that averages the outputs of both architectures. However, this design was never implemented since the inherent computational overhead would make it unattractive [57].

5.7 Rotating Filter Orders

Undoubtedly, a speech signal that has been transmitted across a telephony network will be subjected to non-uniform distortion as a result of channel noise. This

is simply because the contaminating noise signal does not have constant power across its spectrum. Consequently, certain speech segments will be distorted by high-energy noise, whereas other portions will be affected by low-energy noise. Therefore the extent of smoothing must be adjusted dynamically depending on local noise content in a speech signal. Due to this requirement, a so-called parametric voice activity detection (VAD) module was implemented. The parametric VAD returns a score which is indicative of the noise content in a given frame.

Unfortunately, the parametric VAD was later found to introduce excessive computational overhead in addition to sporadic error (i.e. the occasional misclassification of frames), and was therefore discontinued. **Moreover, this VAD also relied on the assumption of long-term stationarity for the contaminating noise function [73], which was considered by the author to be somewhat unrealistic**. As a result, the author decided to make use of frame energies instead. The energy in a given frame was calculated simply as:

$$FE = 10 * log(\sum_{i=0}^{N} x_i^2) \qquad (5.1)^1$$

where FE is the frame energy, x_i is the ith sample, and the summation runs from 0 to 319 (i.e. 320 samples in a frame, corresponding to a sampling rate of 16 kHz and a frame interval of 20 ms). Clearly, frames that have high energies (i.e. presumably high SNR, or clean frames) need not be filtered, whereas heavily-contaminated frames might need to be filtered with a low-order polynomial (i.e., a low-fidelity) filter. For instance, a frame that consists entirely of (zero-mean) noise could be smoothed with a $0th$ order polynomial (i.e. to signify absence of speech), while a frame that is virtually unaffected ought to be "filtered" using a $6th$ order polynomial (i.e., no filtration, assuming an SFL of 7 samples) since a lower-order filter would compromise fidelity and thus induce distortion. **Thus, it**

[1]This equation was taken from the code provided by the author's supervisor, Dr D.J. Mashao

is clear that unconditional smoothing is not desirable as it might introduce unwanted distortion and thus diminish performance. Indeed this was found to be true in a few cases. That is, in a few cases, clients who were correctly verified using the baseline architecture were rejected by the PA algorithm even though the latter's overall performance was superior.

5.8 Discussion of Results

The relatively small reduction in EER of 9.2 % is probably due to the fact that a significant part of the noise in a telephony channel is Gaussian since non-linear filters are most effective at reducing non-Gaussian noise. Alternatively, the filter that was used might not be the most effective one at reducing this type of noise. Unfortunately, the author was not able to find any literature in which identical work had been done using a different non-linear filter. Moreover, the primary source of distortion in a telephone channel is convolutional noise [8][69], which cannot be reduced using a simple filter (linear or non-liniear). This means that even the most effective non-linear filter would still provide only a slight improvement in performance. **Nevertheless, the rationale behind using the PA filter was instigated by the fact that, to the author's knowledge, no noise compensation algorithm had specifically been designed to suppress non-Gaussian noise in a telephone network for speaker recognition systems using a PA-based filter**.

Furthermore, the author did not implement an impulse noise detection module because it would introduce excessive computational overhead but offer only a slight improvement in performance since the issue of convolutional distortion would still need to be addressed in any case. In addition, the author believes that this problem (i.e. convolutional distortion) is beyond the scope of a Master's thesis simply because of the sheer amounts of time and research effort that would be required.

Finally, the EER results in shown in Table 5.3 fluctuate quite considerably. The author believes that this is due to the fact that there were much fewer FRR bids than FAR bids (i.e. 168 versus 28056). This disparity was dictated by the fact that there is generally very little data in NTIMIT (i.e. only 10 utterances per speaker). Had there been many utterances (e.g. in [31], where 9 utterances were used for each verification bid), the author would have used far more FRR bids (perhaps 6 per speaker, which would increase the number of FRR transactions to 1008 in total). This would mean that the distribution of true client scores is more uniform (as opposed to sporadic). **However, the slightly better average EER corresponding to the PA algorithm is corroborated by the associated D' averages**.

5.9 Summary

This chapter essentially presented the simulation results of both the baseline architecture and the proposed algorithm. Using the EER and the D' evaluation metrics, it was found that the proposed algorithm consistently outperformed the baseline architecture, albeit by a narrow margin. In addition, the baseline architecture's performance was found to be very competitive compared to other architectures reported in literature. Finally, the PA algorithm was found to be considerably more computationally intensive than the baseline system.

Chapter 6

Conclusion

This chapter presents a concise summary of the work that was done in this project. In addition, an appraisal of both cohort- and world-model based anti-speaker modelling methodologies is presented, as well as a realistic prognosis for SV in general.

6.1 What Was Achieved

As outlined in section 1.3, the first objective of this thesis was to design and implement a baseline text-independent SV architecture whose performance (expressed in terms of the EER), is comparable to those of competing architectures reported in literature. This objective was accomplished successfully since even the baseline architecture's EER was better than all others (tested on NTIMIT) that the author found while conducting this research.

Furthermore, results obtained using the PA algorithm demonstrate its superiority compared to the baseline arhitecture. In order to ascertain the validity of these results, three simulations were performed using both architectures, and the resultant

EER values were averaged. Based on these results, it is obvious that the PA algorithm demonstrably outperforms the baseline architecture. This improvement was further qualified by computing D' figures for both archictures. As expected, client and impostor score distributions corresponding to the PA algorithm were found to be more "separable" than those obtained using just the baseline architecture, thus confirming the superiority of the proposed algorithm.

However, the baseline architecture was shown to be much faster than the PA architecture. At any rate, the PA algorithm is still fast enough for it to be used in real-time verification transactions. The difference in speed is only really felt during training where it trails by at least several hours for a population of 168 speakers.

6.2 The EER as an Evaluation Metric

Since the EER can only be computed using actual FAR and FRR values, it is obvious that calculating the EER of a fielded SV system that has not been tested before would not be straightforward. However, it should be possible to extrapolate the EER using calibration data, for instance. In other words, if there are, say, 15 available training utterances, 12 could be used to create speaker and world models, and the remaining 3 used as *pseudo* testing material in order to estimate the EER. Nevertheless, the FRR and the FAR need not be equal in reality. In banking applications, for instance, it might be preferable to have an FRR of say, 3 %, and an FAR of close to 0 %. Furthermore, quoting an EER value without specifying a) the amounts of training and testing speech, and b) whether universal or speaker-specific thresholds were used is a bit pointless. Thus an SV system that has an EER of 3.0 % obtained using 2 hours of training speech and 10 minutes of testing speech as well as speaker-dependent decision thresholds is not necessarily superior to one that has an EER of 3.2 % corresponding to 30 minutes of train-

ing speech and 5 minutes of testing data, without the use of speaker-dependent thresholds.

6.3 A Prognosis for Speaker Verification

Although speaker verification is still an emerging technology, it is doubtful whether future SV architectures will ever to able to function perfectly a) in diverse acoustic environements, b) across mobile and telephony networks, and c) despite excessive speaker variability, such as when a male child reaches puberty and his voice deepens unpredictably. The problem with a) above is that some environments could include the voices of other speakers, such as in a busy airport terminal. In this context, it would obviously be very difficult to differentiate the speaker's voice from those of other speaker's in his immediate vicinity.

In addition, the problem of telephone channel distorted speech is compounded by the fact that it is extremely difficult to isolate the source of distortion. In other words, higher EER values associated with telephony speech (as opposed to those corresponding to clean speech) could be due to mismatched handsets, additive noise, impulsive noise, convolutional distortion and cross-talk. Thus it is undoubtedly difficult to design a single architecture that is capable of effectively compensating for all these artefacts.

Furthermore, contemporary anti-speaker modelling methodologies do not consider the fact that, in reality, impostors would most likely modulate their voices when targeting specific victims. **For instance, a Spanish-speaking male impostor would most likely put on a fake accent and pitch when targetting a German-speaking female client, provided that her identity is known to him.** Thus it is highly probable that a given client could have more 'cohorts' in reality than might appear to be the case. This is especially disconcerting because most

clients would speak naturally during training (assuming a closed-set scenario), making it extremely difficult to accurately identify the cohort set for a particular speaker. In addition, impostor attacks could also originate from external users (i.e. open set verification), which further undermines the cohort methodology.

Likewise, the world model methodology is also vulnerable to 'adaptive' impostors. This is simply because an impostor who is good at impersonating other people could speak naturally during training and adapatively modulate his voice when targetting a specific victim. **Thus this impostor would in essence be presenting data that the system had not seen before, meaning that the range of alternative speech that is used to generate world (or UBM) models would be incomplete**.

On a positive note, it is unlikely that a potential impostor would both sound like the victim and know the victim's code (assuming a code-based identity-claim system). Therefore, such an impostor might be good at impersonating other speakers, but it would be pointless if he does not have the required code. Furthermore, it is a bit unfair to expect SV to work perfectly when no other biometric verification system is perfect. For instance, fingerprint recognition might fail if one's thumb is lacerated or disfigured (e.g. a by a wound). Moreover, it is not uncommon for individuals to have multiple passports or counterfeit identity documents.

In light of all these considerations, the author believes that truly successful far-field SV architectures will a) comprise some sort of error correction capability, and b) function as auxiliary identity authentication algorithms and not as stand-alone verification architectures. In the context of mobile telephony, future handsets could compute features like the MFCC and use forward error correction (FEC), such as convolutional encoding. However, this would increase the amount of bandwidth required for mobile telephone networks. A possible solution would be to use more efficient modulation schemes such as EDGE (Enhanced Data rates

for Global Evolution) in order to keep bandwidth at a minimum. EDGE provides a far more efficient usage of the radio spectrum than conventional modulation schemes [EDGE]. An advantage of generating features locally within the handset and using spontaneous error correction to ensure their integrity is that channel effects would be diminished. Moreover, features would no longer have to be computed at the remote end, meaning that buffers would be emptied much faster. This also means that the *mean call holding time* (MCHT) for a typical verification transaction would be reduced, which would be very useful if clients had to pay for the call themselves (assuming no 800 number service).

An additional advantage of using mobile handsets (as opposed to the *plain old telephony system*, or POTS) is that clients are more likely to always use the same handset. This would thus address the problem of mismatched handsets, even if spontaneous error correction were not used. Lastly, ordinary telephone lines may be "tapped" almost effortlessly, which means that it would be fairly easy for criminals to either record or synthesize a client's speech as well as learn their secret code. However, mobile handsets cannot be tapped as easily. Furthermore, this country is especially fortunate because of the extent of penetration that mobile telephony has achieved here.

Bibliography

[1] http://anacardier.eecs.tulane.edu/documentation/htkbook/node62.html. Last Accessed 28 April 2004.

[2] http://bulsai.kaist.ac.kr/sllhp02/tech. Last accessed 28th November 2003.

[3] http://home.earthlink.net/ philipshinn/svsite/sv.htm. Last accessed 28 April 2004.

[4] http://web.njit.edu/ ang/papers/ip99.htm. Last accessed 31st May 2004.

[5] http://www-psych.stanford.edu/ lera/psych115s/notes/signal.html. Last accessed 5th December 2003.

[6] http://www2.mdanderson.org/app/ilya/nonlinear.htm. Last accessed 31st May 2004.

[7] http://www.cepstrum.co.jp/1about.htm. Last accessed 31st May 2004.

[8] http://www.eie.polyu.edu.hk/ mwmak/speakerversys.htm. Last accessed 31st May 2004.

[9] http://www.iis.sinica.edu.tw/lib/gauss.html. Last accessed 31st May 2004.

[10] http://www.jmarkowitz.com/ask.html. Last accessed 31st May 2004.

[11] http://www.kpn-telecom.nl/cave/project.html. Last accessed 31st May 2004.

[12] http://www.owlnet.rice.edu/ elec539/projects99/bach/proj2/blind/bd.html. Last accessed 31st May 2004.

[13] http://www.rpi.edu/ zouj/wiener/filter.html. Last accessed 31st May 2004.

[14] http://www.stfx.ca/people/pkeizer/stats/lombardeffect.htm. Last accessed 31st may 2004.

[15] www.ispeak.nl/prfhtm/node12.html. Last accessed 31st May 2004.

[16] *A Description of HTK Application Programming Interface*, chapter 11, page 105. Entropic Limited, Apr. 1997.

[17] N.T. Baloyi. A comparison of features for large population speaker identification. Master's thesis, University of Cape Town, Sep. 2000.

[18] S. Bhattacharyya, T. Srikanthan, and A. Panda. Global background model approach for embedded speaker verification systems. *Proceedings of International Symposium on Communication Systems, Networks and Digital Signal Processing*, pages 383–386, 2002.

[19] F. Bimbot, M. Blomberg, L. Boves, D. Genoud, H.-P Hutter, C. Jaboulet J. Koolwaaij, J. Lindberg, and J.-B. Pierrot. An overview of the cave project research activities in speaker verification. *In Proc. La Reconnaissance du Locuteur et ses Applications Commerciales et Criminalistiques (RL2AC)*, pages 215–220, 1998.

[20] H. Bourlard and N. Morgan. Speaker verification: A quick overview. Technical report, Dall Moll Institute for Perceptual Artificial Intellingence (IDIAP), Aug. 1998.

[21] J.P. Campbell. Speaker recognition: A tutorial. *Proceedings of the IEEE*, Vol. 85(9):1437–1462, Sep. 1997.

[22] K. Chen. A hybrid score measurement for hmm-based speaker verification. *ICASSP'99*, 1:317–320, Mar. 1999.

[23] K. Chen. Towards better making a decision in speaker verification. *PATTERN RECOGNITION*, Vol. 36(No. 2):329–346, 2003.

[24] G. Chollet, J.-L. Cochard, A. Constantinescu, C. Jaboulet, and P. Langlais. Swich french polyphone and polyvar telephone speech databases to model inter- and intra-speaker variability. Technical report, IDIAP, 1996.

[25] M.J. Coker and D.N. Simkins. A nonlinear adaptive echo-canceller. *In Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 470–473, 1980.

[26] R.W. Daniels. *An Introduction to Numerical Methods and Optimization Techniques*, chapter 4, pages 86–89. Elsevier North-Holland, 1978.

[27] M. El-Maliki and A. Drygajlo. Missing features detection and handling for robust speaker verification. *In Proceedings of the 6th European Conference On Speech Communication And Technology*, pages 975–978, 1999.

[28] H. Ezzaidi, J. Rouat, and D.O'Shaughnessy. Towards combining pitch and mfcc for speaker identification systems. Technical report, Universit'e du Qu'ebec, 2001.

[29] M. Falcone and A. Gallo. The "siva" speech database for speaker verification: Description and evaluation. *International Conference on Spoken Language Processing (ICSLP'96)*, pages 1902–1906, 1996.

[30] The Centre for Communication Interface Research. Large scale evaluation of speaker verification technology. Technical report, University of Edingburgh, May 2000.

[31] The Centre for Communication Interface Research. Evaluation of nuance v7.04 speaker verification performance on the dialogues spotlight uk english database. Technical report, University of Edinburg, 2001.

[32] M.E. Forsyth and M.A. Jack. Discriminating semi-continuous hmm for speaker verification. *Proceedings of the International Conference On Acoustics, Speech, And Signal Processing*, Vol. 1:313–316, Apr. 1994.

[33] W.A. Frank. An efficient approximation to the quadratic volterra filter and its applications in realtime loudspeaker linearization. *Signal Processing*, Vol. 45:97–113, 1995.

[34] A. A. Gasteratos. *Development And Hardware Implementation Of New Techniques For Non-linear Image Processing*. PhD thesis, Democritus University of Thrace, Greece, 1998.

[35] A.T. Georgiadis and B. Mulgrew. A map equaliser for impulsive noise environments. Technical report, University of Edinburgh (Department of Electrical and Electronic Engineering).

[36] S.J. Godsill and P.J.W. Rayner. Robust treatment of impulsive noise in speech and audio signals. *Bayesian Robustness*, pages 315–326, 1996.

[37] G. Gravier and G. Chollet. Comparison of normalization techniques for speaker verification. *Proc. Speaker Recognition and its Commercial and Forensic Applications*, pages 97–100, Apr. 1998.

[38] M. Bett R. Gross, H. Yu, X. Zhu, Y. Pan, J. Yang, and A. Waibel. Multimodal meeting tracker. Technical report, Carnegie Mellon University (Interactive Systems Laboratories), 2000.

[39] W. Henkel and T. Kessler. A wideband impulsive noise survey in the german telephone network. *Statistical Description and Modeling*, Vol. 48(6):277–288, 1994.

[40] H. Hermansky and N. Malayath. Speaker verification using speaker-specific mappings. *Speaker Recognition And Its Commercial And Forensic Applications*, pages 111–114, Apr. 1998.

[41] H. Hermansky and N. Malayath. Speaker verification using speaker-specific mappings. *Speaker Recognition And Its Commercial And Forensic Applications*, pages 111–114, Apr. 1998.

[42] Aapo Hyvarinen. http://www.cis.hut.fi/aapo/papers/ncs99web/node10.html. Last accessed 31st May 2004.

[43] H.M. Kim and B. Kosko. Fuzzy filters for impulsive noise. *Fuzzy Engineering*, pages 251–257, 1997.

[44] G. Klasmeyer, T. Johnstone, T. Banziger, and C. Sappok. Emotianl voice variability in speaker verification. *International Speech Communication Association*, pages 213–218, Aug. 2000.

[45] R.L. Klevans and R.D. Rodman. *Voice Recognition*, chapter 1, page 4. Artech House, 1997.

[46] Johan Koolwaaij. http://www.ispeak.nl/prfhtm/node13.html. Last accessed 31st May 2004.

[47] Yatin Kulkarni and Anil Jain. http://biometrics.cse.msu.edu/speaker.html. Last accessed 31st May 2004.

[48] J. Larsen. *Design of Neural Network Filters*. PhD thesis, Technical University of Denmark, 1993.

[49] Q. Le and S. Bengio. Client dependent gmm-svm models for speaker verification. Technical report, IDIAP, 2003.

[50] X. Li, E. Chang, and B. q. Dai. Improving speaker verification with figure of merit training. *In Proceedings of the International Conferences on Acoustics, Speech and Signal Processing (ICASSP)*, pages 693–696, 2002.

[51] Aranz Limited. http://aranz.com/research/modelling/theory/smoothrbf.html. Last accessed 31st May 2004.

[52] J. Lindberg, J. Koolwaij, H. P. Hutter, D. Genout, J. B. Pierrot, M. Blomberg, and F. Bimbot. Techniques for a priori decision threshold estimation in speaker verification. *Speaker Recognition and its Commercial and Forensic Applications*, pages 89–92, Apr. 1998.

[53] M. W. Mak. Text-independent speaker verification over a telephony network by radial basis functions. *In Proceedings of the International Symposium on Multi-Technology Information Processing (ISMIP'96)*, pages 145–150, 1996.

[54] D.J. Mashao. *Computations and Evaluations of an Optimal Feature-set for an HMM-based Recognizer*. PhD thesis, Brown University, May 1996.

[55] Hemant Misra, Shajith Ikbal, and B. Yegnanarayana. Speaker-specific mapping for text-independent speaker recognition. *Speech Communication*, Vol. 39:301–310, Feb. 2003.

[56] P. J. Moreno and R. M. Stern. Sources of degradation of speech recognition in the telephone network. *In Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1:104–109, 1994.

[57] L. B. Mothae and D.J. Mashao. Using a polynomial approximation-based impulse noise suppression algorithm for robust speaker verification. *Fourteenth Annual Symposium Of The Pattern Recognition Association of South Africa*, pages 85–89, Nov. 2003.

[58] L.B. Mothae and D.J. Mashao. Speaker verication on an identification system using parametric feature-sets. *Pattern Recognition Association of South Africa*, pages 56–59, 2002.

[59] M. Newman, L. Gillick, Y. Ito, D. McAllaster, and B. Peskin. Speaker verification through large vocabulary continuous speeh recognition. *Proc. ICSLP-96*, pages 2419–2422, 1996.

[60] G. Nokas, E. Dermatas, and G. Kokkinakis. Robust speech recognition in noisy reverberant rooms. *SPECOM-97*, pages 105–108, 1997.

[61] J. Ortega and J. Gonzalez Rodriguez. Overview of speech enhancement techniques for automatic speaker recognition. *International Conference on Spoken Language Processing*, Vol. 2:929–932, Oct. 1996.

[62] D. Petrovska. Polycost: A telephone speech database for speaker recognition. *In Proceedings of La Reconnaissance du Locuteur et ses Applications Commerciales et Criminalistiques (RL2AC)*, pages 211–214, 1998.

[63] JIRI PORUBA. http://www.electronicsletters.com/papers/2001/0021/paper.asp. Last accessed 31st May 2004.

[64] I. Potamitis, N. Fakotakis, and G. Kokkinakis. Map spectral estimation for on-line noise compensatoin of time trajectories of spectral coefficients. Technical report, Wire Communications Laboratory, Electrical and Computer Engineering Dept., University of Patras, 2001.

[65] Ilyas G. Potamitis. *Speech Enhancement Techniques in the Context of Automatic Speech Recognition*. PhD thesis, Department of Electrical and Computer Engineering, Speech and Language Technology Group, University of Patras, 2001.

[66] P. Premakanthan and Wasfy B. Mikhael. Speaker verification / recognition and the importance of selective feature extraction : Review. Technical report, University of Central Florida (Department of Electrical Engineering), 2001.

[67] D. A. Reynolds. Automatic speaker recognition using gaussian mixture models. *MIT Lincoln Laboratory Journal*, Vol. 8(2):173–192, 1996.

[68] D.A. Reynolds. Speaker identification and verification using gaussian mixture speaker models. *Speech Communication*, pages 91–108, Aug. 1995.

[69] D.A. Reynolds. The effects of handset variability on speaker recognition performance: Experiments on the switchboard corpus. *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'96)*, pages 113–116, May 1996.

[70] D.A. Reynolds. Corpora for the evaluation of speaker recognition systems. *ICASS*, 1999.

[71] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, Vol. 10:19–41, 2000.

[72] N. Sakai. Jacobian joint adaptation to noise, channel and lombard effect. Master's thesis, Japan Advanced Institute of Science and Technology, 2002.

[73] C. Sanderson. Speech processing and text-independent automatic person verification. Technical report, IDIAP, Dec. 2002.

[74] J.C. Segura, M.C. Benitez, A. de la Torre, and A. Rubio. Feature extraction combining spectral noise reduction and cepstral histogram equalization for robust asr. *Proc. ICSLP'2002*, pages 225–228, Sep. 2002.

[75] M. Sigmund. Speaker recognition - identifying people by their voices. Technical report, Brno University of Technology, Faculty of Engineering and Computer Science, Institute of Radio Electronics, 2000.

[76] J.C. Stapleton and S.C. Bass. Adaptive noise cancellation for a class of nonlinear, dynamic reference channels. *IEEE Trans. on Circuits and Systems*, Vol. 32(2):143–150, Feb. 1985.

[77] Todd Veldhuizen. http://osl.iu.edu/ tveldhui/papers/mascthesis/node23.html. Last accessed 31st May 2004.

[78] Q. Yang and J.-P. Martens. On the importance of exception and cross-word rules for the data-driven creation of lexica for asr. *Proceedings of Pro RISC*, pages 589–593, 2000.

[79] D. Zhang and Z. Wang. Impulse noise removal using polynomial approximation. *Optical Engineering*, 37(4):1275–1282, 1998.