T2R2 東京科学大学 リサーチリポジトリ
Science Tokyo Research Repository

論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Robust Background Models for Speaker Verification
著者(和文)	ﾋﾞｽﾜｽｻﾝｷﾞｰﾀ
Author(English)	SANGEETA BISWAS
出典(和文)	学位:博士（学術）, 学位授与機関:東京工業大学, 報告番号:甲第10048号, 授与年月日:2016年1月31日, 学位の種別:課程博士, 審査員:篠田　浩一,亀井　宏行,徳永　健伸,藤井　敦,石田　貴士
Citation(English)	Degree:, Conferring organization: Tokyo Institute of Technology, Report number:甲第10048号, Conferred date:2016/1/31, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Powered by T2R2 (Science Tokyo Research Repository)

Robust Background Models for Speaker Verification

Sangeeta Biswas
Supervised by Professor Koichi Shinoda

Department of Computer Science
Graduate School of Information Science and Engineering
Tokyo Institute of Technology

Dissertation submitted to the Tokyo Institute of Technology
for the degree of Doctor of Philosophy

2015

Abstract

An automatic speaker verification system (ASVS) is used to determine whether a claim about the identity of a person is $true$ or $false$ by analyzing the person-specific characteristics extracted from speech. In a text-independent ASVS, the lexical contents of speech are allowed to be varied from time to time. This kind of system has a great demand in surveillance and forensic applications. In the last two decades, short utterances and inter-session variability have been considered two important obstacles for achieving high verification accuracy. In this thesis, we deal with these two obstacles by improving background models named structural maximum-a-posteriori (SMAP) tree and probabilistic discriminant analysis (PLDA) model. These background models have large impact on the system's performance.

When utterances are very short, SMAP adaptation is good for reliable parameter estimation of speaker-specific models. In SMAP adaptation, a tree structure obtained by clustering Gaussian components offers a convenient way to capture the hierarchical structure of the acoustic space of the human voice. Different speakers may have different acoustic spaces depending on factors such as their language, accents or pronunciation particularities. We propose to use a set of trees instead of using a single tree, assuming that each speaker will find its appropriate tree. We define the set of trees as an acoustic forest. Combining decisions of many ASVS using different tress in the acoustic forest gives robustness against enrollment data variants.

For inter-session variability, two utterances with the same lexical content may sound differently even though they are spoken by the same speaker. Recently the PLDA model has become one of the state-of-the-art inter-session variability compensation models. The standard approach is to train a PLDA model by using all available data without paying attention that irrelevant data may deteriorate performance of an ASVS. We notice that selecting k nearest neighbors, we can remove irrelevant training data and improve the system performance. In order to avoid the difficulty of optimizing k on a development set, we propose flexible k-Nearest Neighbors (fk-NN).

By conducting experiments on the text-independent ASVS using conversational telephone speeches, we show the effectiveness of our proposed methods for improving background modeling which ultimately improves verification accuracy.

Acknowledgment

I would like to express my sincere thanks to

- my adviser, Professor Koichi Shinoda, for providing me all the necessary facilities for my research and for being available to discuss about problems as a 24/7 service center.
- Professor Sadaoki Furui for his attention and advice.
- Johan Rohdin for his patience to be with me from the beginning to the end of this work, especially for his sleepless nights solving my problems.
- my thesis committee, especially Associate Professor Atsushi Fujii, for insightful comments and encouragement that helped improve this thesis.
- our secretaries Yoko Kawakami and Yoko Takashina for official help.
- MEXT (Ministry of Education, Culture, Sports, Science, and Technology-Japan) for financial support.
- all SHINODA laboratory members for being my friends and building a family environment for me.
- all my old Japanese friends (whom I met either very early in the morning or very late at night) for their smiles and proving that language does not matter for friendships.
- all beautiful flowers, plants, trees and birds surrounding Tokyo Tech campus for giving me a reason to be cheerful.
- my special friends, Berrin, Francesca, Ryan, and Xie for cheering me up whenever I felt down.
- my friends Kishan, Tommi, Mengxi and Zhuolin for helping me finish this thesis.
- my beloved father, mother, uncle, sister and brothers for their unconditional love and support which helped me keep going.

Dedication

I would like to dedicate this thesis to my father, Subash Biswas, and Johan Rohdin.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we give a brief introduction of an automatic speaker verification system and mention the focus of this dissertation. The organization of this chapter is as follows: In Section 1.1, some differentiating points between an automatic speaker verification system (ASVS) and an automatic identification system are mentioned. Section 1.1.1 gives a comparative study between an ASVS and other biometric authentication systems. The performance issue, applications and obstacles for getting high verification accuracy of an ASVS are described in Section 1.1.2, Section 1.1.3, and Section 1.1.4, respectively. Section 1.2 is about our research direction and contribution. The organization of the rest of this dissertation is given in Section 1.3.

1.1 Fundamentals of Speaker Verification

Automatic speaker verification is a task to determine whether a claim about the identity of a person is $true$ or $false$ by analyzing the person-specific characteristics extracted from speech without any manual interference. The system which performs this task is known as an *automatic speaker verification system* (ASVS). The person who provides speech and claimed ID to an ASVS is called a *speaker* or *user*, and the claimed identity is called a *target speaker*. When a speaker claims him/herself as a *target speaker* falsely, then that speaker is called an *impostor*. On the other hand, when the *target speaker* is the one who a speaker claims to be, then that speaker is called *true speaker* or *client*.

An ASVS is different from an *automatic speaker identification system* (ASIS) for few reasons. An ASIS gives an identity to a person by estimating the similarity of that person's voice with n registered speakers' voices and choosing the most similar voice's ID as that person's ID. Therefore, it is a 1-to-n comparison task whereas a speaker verification task is a 1-to-1 comparison task because it matches the similarity of a speech segment with a particular speaker's voice. An ASIS establishes a user's identity (ID) answering the question "*Who am I?*", whereas an ASVS answers to the question "*Am I whom as I claim I am?*". In an ASIS a user does not provide any other information except speech, whereas in an ASVS a user needs to provide the ID of the claimed speaker along with the speech.

An ASVS has two phases: training phase and authentication phase. When users are restricted to utter the same speech both in the training phase and the authentication phase, an ASVS is called a *text-dependent* system. It is mainly used for co-operative users (who willingly participate in the training and the authentication phase, or who are aware about the authentication process) in a controlled environment. Few seconds of speech is enough to obtain performance that is good enough for commercial applications. On the other hand, when the lexical contents of speech are allowed to be varied from time to time, an ASVS is called a *text-independent* system. It can be used for users who are unaware about the authentication process. Therefore, this kind of system has a great demand in surveillance or forensic applications. Long utterances are necessary in the training phase in order to capture the long-term statistical characteristics of the target speaker's speech. When a system can generate synthesized speech or text for users to inform what to utter, a text-independent ASVS is a good choice for avoiding the impostor attack done by using the recorded speech. This kind of text-independent ASVS is known as *text-prompted ASVS*.

1.1.1 Comparison with Other Biometric Systems

The individuating characteristic of a human being carried by speech is known as *voice biometric*, and an ASVS is sometimes called a *voice authentication system*. Each person generally has a distinctive voice due to unique physiological properties of his/her speech production system along with his/her speaking style which includes accent, rhythm, pronunciation, etc. There-

fore, human voice is considered as a combination of physiological and behavioral biometrics. The physiological part is invariant for an individual, but the behavioral part changes over time due to age, medical conditions (e.g., cold), and emotional state (e.g., stress), etc. [71]. A brief comparison between a voice authentication system and other biometric authentication systems based on the following seven factors pointed out by Jain et al. [71] is given in Table 1.1.

1. universality: the size of the population that have the biometric.
2. distinctiveness: how well the biometric separates individuals from each other.
3. permanence: how stable the biometric is over time.
4. collectivity: how quantitatively the biometric is measured.
5. performance: to what extent accurate and speedy results can be obtained by the biometric.
6. acceptability: to what extent people are willing to use the biometric.
7. circumvention: how easily the system can be fooled using fraudulent actions.

Even though the distinctiveness of an ASVS is low and it is easy to be fooled by fraudulent actions, since 1960s [108, 93, 123, 97] until now ASVSsa' have been drawing huge attention from researchers because voice is the only biometric that is based on the acoustic information, whereas most other biometrics like *face*, *retina*, *fingerprint*, *hand geometry*, *gait*, etc., are image-based. For the remote-access transactions, voice is a good choice because of the following reasons:

1. Speech is a natural communication media of the human being. Most of the people do not need any extra effort to speak. Therefore, the acceptability of voice as a biometric to the user is very high. Even voice is more acceptable than face (another highly acceptable biometric for authentication) to the users who are unwilling to show their face due to religion or customs.
2. Speech can be captured easily with simple transducers and recording devices. Comparing to cameras for capturing *facial features* or scanners for capturing *iris*, or *palmprint*, voice capturing devices are less expensive.

Table 1.1: Comparison of various biometric technologies based on the perception of Jain et al. [71]. Universality, Distinctiveness, Permanence (Stability), Collectivity, Performance, Acceptability, and Circumvention (Evasion) are denoted by 'U', 'D', 'S', 'C', 'P', 'A' and 'E', respectively. High, Medium, and Low are denoted by 'H', 'M', and 'L', respectively.

Biometric Identifier	U	D	S	C	P	A	E
DNA	H	H	H	L	H	L	L
Ear	M	M	H	M	M	H	M
Face	H	L	M	H	L	H	H
Facial thermogram	H	H	L	H	M	H	L
Fingerprint	M	H	H	M	H	M	M
Gait	M	L	L	H	L	H	M
Hand geometry	M	M	M	H	M	M	M
Hand vein	M	M	M	M	M	M	L
Iris	H	H	H	M	H	L	L
Keystroke	L	L	L	M	L	M	M
Odor	H	H	H	L	L	M	L
Palmprint	M	H	H	M	H	M	M
Retina	H	H	M	L	H	L	L
Signature	L	L	L	H	L	H	H
Voice	M	L	L	M	L	H	H

3. For well-established telecommunication networks, speech can be transferred without any extra system setup.

1.1.2 Performance

According to a presentation entitled "On the Deployment of Speaker Recognition for Commercial Applications:Issues & Best Practices" in International Biometrics Consortium, 2003, by Larry Heck, the director of speech R&D at Nuance Communications, the error rate of an ASVS was:

1. Text-dependent ASVS using
   - clean data, single microphone, and large amount of training & authentication data: **0.1%**

- telephone data having digit strings, multiple microphones and small amount of training data: **1%**

2. Text-independent ASVS using
    - conversational telephone data, multiple microphones and moderate amount of training data: **10%**
    - military radio data having read sentences, multiple radios & microphones and moderate amount of training data: **25%**

Since then the performance of an ASVS specially of a text-independent ASVS has been improved a lot. According to a report by Greenberg et al. in 2014 [49], the error rate of a text-independent ASVS is **2-5%**.

1.1.3 Applications

Rapid growth of mobile communications has increased the demand of an ASVS system. These days huge amount of information is exchanged between two parties in telephone conversations, including between criminals. From blackmailing to sending commands to terrorists for malicious activities, criminals and leaders of terrorists are using speech disguising their face or other biometrics. Sometimes some videos are prepared by combining different person's voice with the face of a well-known leader or a terrorist in order to mislead his/her followers. In such scenarios, no other biometric authentication system can be used rather than an ASVS. In 2003, the CIA (Central Intelligence Agency), the NSA (National Security Agency) and Swiss IDIAP (Dalle Molle Institute for Perceptual Artificial Intelligence) used an text-independent ASVS to analyze so called Osama Bin Laden's fake tapes. Since 1980, an ASVS has been used for other commercial purposes rather than forensics, such as online banking, network access-control, automated customer services, etc. Some successful applications of ASVSs are given below:

- **SAIVOX**: operated by the Spanish Guardia Civil, a law enforcement agency. It is used to find out criminals using voice samples by comparing with around 3,500 recordings linked with well-known criminals and certain types of crime.

- **PerSay VocalPassword**: operated by Vodafone Turkey. It is used to secure self-service applications such as GSM Personal Unlocking Key

reset and access to Vodafone Call Centers.

- **VoicePassport**: developed by Japanese Animo company. It is used for secure access to personal databases, logging into e-learning systems, telephone banking, and other telephony services.
- **BATVOX**: developed by Spanish Aginto company. It is designed for criminal identification experts and scientific police to perform speaker verification and compile expert reports as evidence in court. It is used in court in more than 35 countries world wide as the de facto standard Voice Biometrics forensic tool.
- **Lenovo A586 voice unlock**: jointly created by Baidu and the Singapore-based A*STAR Institute for Infocomm Research. It is used to securely lock down an Android phone using user's voice.
- **FreeSpeech**: developed by Nuance company. It is used to verify a caller's identity during the course of a natural conversation for enterprises, interactive service responses (IVRs) service providers and call centers. It transparently retrieves the biometric voice characteristics required for verification within seconds, regardless of what is said, accent, language, or call quality.
- **VocalPassword**: developed by Nuance company. It is used to verify a speaker during an interaction with a voice application such as an IVR or a mobile application. It compares a single repetition of a passphrase that is stored in the system's database.

1.1.4 Challenges

Countless factors greatly destroy the unique characteristics of speech signals and make reliable speaker verification a complicated and challenging task ([111, 54, 33]). Some factors are mentioned below:

- Insufficient Amount of Relevant Speech
  - very short utterances
- Inter-session Variability
  - different microphones, transmission channels in the training and the testing phase

  - speaker's different physical and psychological states (e.g., caught cold)

- Abnormal Speaker States
  - extreme emotional states (e.g., stress or duress)
  - loudness of speech (whispering, normal talking or very loud speaking)
- Intentional Circumvention
  - mimicry, replay, speaker-adapted speech synthesis
- Distorted Speech Signal
  - surrounding environmental noise, transmission channel noise echo, crosstalk

In order to train speaker-specific models robust against the above mentioned factors, we need speech data for all conditions. However, even in the text-dependent ASVS, it is unpractical to ask users to say the same phrase in multiple conditions. In the text-independent ASVS, where the phonemes of the authentication utterance are unknown, many hours of speech data in multiple conditions are necessary in order to cover the effects of all these factors during the training period of speaker-specific models. Therefore, text-independent verification is more challenging than the text-dependent verification. Since the former one is more flexible for the users than the later one and more demanding in forensic applications, in the last two decades a large effort has been given to improve the performance of the text-independent ASVS.

Since 1996, research for text-independent ASVS has been influenced by the NIST *speaker recognition evaluation* (SRE) plan funded by the U.S. Department of Defense. In the NIST SRE plan, the following two factors are considered two important obstacles for achieving high verification accuracy:

- **Challenge-1**: inter-session variability
- **Challenge-2**: short duration of utterance.

The above mentioned two obstacles are not only important for the researchers following the NIST SRE plan, but also for the users of ASVS. Co-operative users prefers to use short speech segment in real applications for convenience. On the other hand, forensic applications need to handle mismatched recordings.

Inter-session variability occurs when there is a mismatch between the training and authentication conditions. For this problem, two utterances with same lexical content may sound differently even though they are spoken by the same speaker. Last one and half decades, most of the researchers working on the NIST SRE plan have been focusing on this problem. The progress of inter-session variability modeling techniques like joint factor analysis (JFA) [78], support vector machine (SVM) with nuisance attribute projection (NAP) [19], or i-vectors with probability linear discriminant analysis (PLDA) [79] is quite impressive when speech segments are around 2.5 minutes long. However, verification accuracy needs to be improved more in order to use a text-independent ASVS in the forensic field.

When speech segments are very short, parameter estimation is not reliable, and therefore, inter-speaker modeling becomes weak. This situation becomes worse when inter-session variability exists. Systems such as JFA, NAP-SVM, PLDA etc., which gave impressive results in long utterances were not successful to solve this problem. Approaches have been proposed to train robust speaker models by tuning few parameters and deal with inter-session variability [100, 35, 130, 81, 103, 76, 77, 84, 74, 75]. For 10-second speech segments, Vogt et al. [130] proposed to use speaker subspace maximum-a-posteriori (MAP) adaptation for factor analysis (FA) modeling, Fauve et al. [34] proposed a well-tuned speech detection front-end for improving frame selection in eigenvoice modeling and Kenny et al. [81] extended JFA to model within-session-variability over a shorter time span.

1.2 Focus of the Dissertation

In this thesis, we deal with the above mentioned two important obstacles in the text-independent ASVS by improving background models. A background model is a model which is trained offline by using data generally not included in the evaluation set. It plays an important role in an ASVS by providing a prior distribution for the parameters of the speaker-specific model, by acting as an alternate model during scoring of a trial, or by providing information about general characteristics of speakers for inter-session variability compensation. Universal background model (UBM) [114], structural maximum-a-posteriori (SMAP) tree [119], total variability matrix [28], NAP matrix [19], and the PLDA model [79] are well known background models.

These models have large impact on the system's performance, however, have obtained little attention from researchers.

The standard approach is to train one background model for the whole set of target speakers using all available data by matching one or two criteria. For example, using *gender*- or *telephone*-specific background models for the whole set of target speakers without paying attention to any other consideration is a well-accepted approach. In this approach, the existence of variations in the set of target speakers and the existence of irrelevant and noisy data in the training dataset of the background models are generally ignored. In this thesis, we pay attention to the target speaker variants and irrelevant data issues for background modeling. We focus on two background models: SMAP tree and PLDA model. The first one is used to mitigate problems caused by short utterances and the second one is used to compensate inter-session variability.

When utterances are very short, SMAP adaptation is good for reliable parameter estimation of speaker-specific models [119]. In SMAP adaptation, a single tree structure is generally used for the acoustic space of all the speakers assuming that the hierarchical structure of the acoustic space can be shared among all the speakers. However, we notice that a single tree structure is not always optimal for modeling the acoustic space of every speaker enrolled to the system. Ideally, different tree structures should be provided for speakers. However, until now no methods for obtaining such trees automatically are known. On the other hand, to find the optimal tree structure for every speaker empirically is computationally expensive when the number of speakers is large, and demands a large amount of data. We propose to use a set of trees instead of using a single tree, assuming that each speaker will find its appropriate tree. We define the set of trees as an *acoustic forest*. Combination of decision of many trees in the acoustic forest gives robustness against enrollment data variants.

Recently, the PLDA model has become one of the state-of-the art inter-session variability compensation technique. It is sensitive to the quality of the training data, $\mathcal{B}$. In order to train a good PLDA model, two conditions need to be fulfilled. First, $\mathcal{B}$ should be plentiful. Multi-channel utterances from several hundred speakers, resulting in several thousands of speech files are typically needed. Second, $\mathcal{B}$ should be relevant; i.e., $\mathcal{B}$ should have similar properties to the evaluation set. Since in very few applications, the prop-

erties of authentication data can be pre-determined, researchers mainly try to match one or two properties of the target speakers with the properties of $\mathcal{B}$ of a PLDA model using meta-data. However, meta-data is not always available for all databases. Beside this, it is not always obvious which properties except gender, telephone type, are important to be matched. Therefore, we adopt data-driven approaches for selecting data from $\mathcal{B}$. We notice that selecting k nearest neighbors we can improve the system performance by removing irrelevant training data of the PLDA model. In order to avoid the difficulty of optimizing k on a development set, we propose *flexible* k-Nearest Neighbors (fk-NN).

1.3 Organization of the Dissertation

Apart from this chapter, this dissertation has five more chapters. **Chapter 2** gives a technical overview of an ASVS and describes the speech databases, protocols and evaluation metrics used in our work. **Chapters 3** discusses previous works on intersession variability compensation techniques and approaches for dealing with short utterances and robust background modeling. **Chapters 4** and **Chapter 5** independently describe our methods to improve the background modeling. While **Chapter 4** tackles the robustness against speaker variants, for which we propose a method to grow acoustic forests, **Chapter 5** handles the robustness against irrelevant training data of the background model, PLDA model, for which we propose flexible k-NN (fk-NN). **Chapter 6** concludes this dissertation and indicates future directions.

Chapter 2

Overview of Automatic Speaker Verification System

In this chapter we give a technical overview of the tasks and modules of an automatic speaker verification system (ASVS). In Section 2.1 we outline the phases of an ASVS. In Section 2.2, datasets generally used in an ASVS are briefly described. The fundamental technologies for features extraction, speaker modeling and adaptation techniques that underlie the work in this thesis are then detailed in Sections 2.3 to 2.5. Finally, the evaluation metrics and significance test used in this thesis are described in Section 2.6 and Section 2.7.

2.1 Phases of an ASVS

As shown in Figure 2.1, an automatic speaker verification system (ASVS) is composed of two phases: (i) a *training* phase and (ii) a *authentication* phase. In the training phase, each speaker who will be target speaker in the authentication phase needs to provide speech segments or utterances in order to train a speaker-specific statistical model. In the authentication phase, a user supplies a speech segment or utterance and claims a speaker identity (target speaker ID), after which the ASVS decides whether the claim is true or false by comparing the speech segment to the model of the claimed identity. There are two steps in the training phase:

1. **Feature Extraction**: In this step, the physiological and behavioral characteristics of the speaker's voice are represented by a sequence

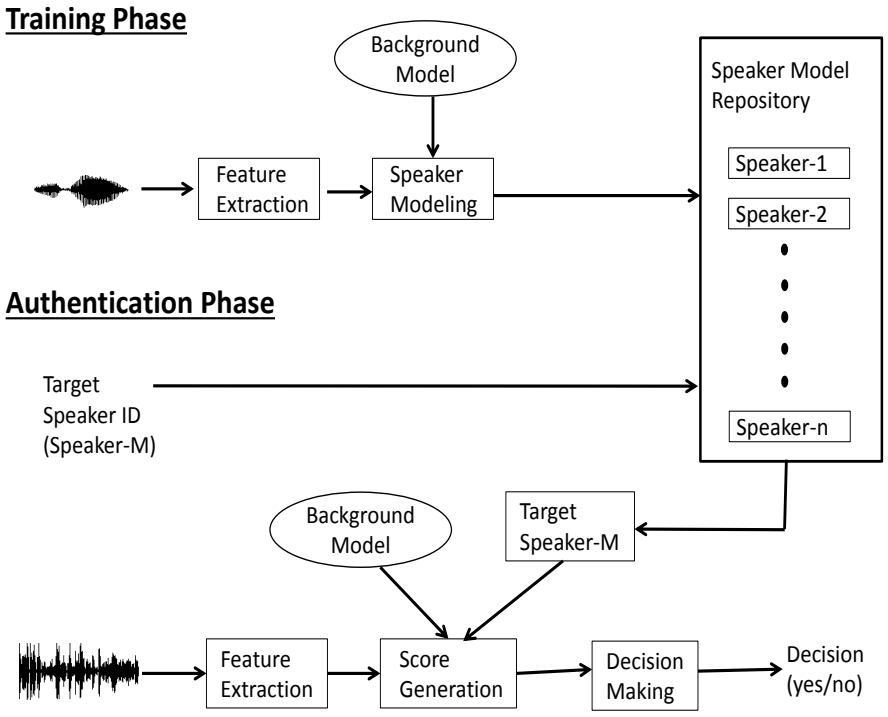

Figure 2.1: Phases of an automatic speaker verification system (ASVS)

of numerical values which are called features (of speech). The features should be designed to capture much of the information about speaker identity in the speech signal as well as being compact and robust against noise etc.

2. **Speaker Modeling**: Speaker models are made for every speaker by statistical modeling of the features extracted from the training speech.

Along with the feature extraction step, the authentication phase of an ASVS has the following two steps:

1. **Score Generation**: In this step, a score is generated for each claim. A score is a numerical descriptor of the speech segment supplied by the user being generated by the claimed speaker (target speaker) model. A good speaker model gives a higher score for the claim of a target speaker than the claim of an impostor.

2. **Decision Making**: Comparing the score with a $threshold$ value, the speaker verification system decides whether the claim of an identity is $true$ or $false$ as follows:

$$\mathcal{D}_x = \begin{cases} True & \text{if} \quad \mathcal{S}_x \geq \Theta, \\ False & \text{if} \quad \mathcal{S}_x < \Theta \end{cases} \tag{2.1}$$

where $\mathcal{D}_x$ is the decision about claim x, $\mathcal{S}_x$ is the score for the claim x, and Θ is a threshold value. If the Θ is set to very high, the system will not do any mistake to take decision about *False* claims from impostors, however it will reject many *True* claims from clients. Similarly, if Θ is very low, many $False$ claims from impostors will be accepted as *True* claims. Two common ways for setting the value of Θ are based on the *equal error rate* (EER) and the *minimum detection cost function* (MinDCF or C^{min}) estimated on a development set (See Section 2.6).

In both phases of an ASVS, background models play important roles. They are generally trained offline. In the training phase of an ASVS, a background model provides a prior distribution for the parameters of speaker-specific models. In the authentication phase, it acts as an alternate model during scoring of a trial, or provides information about general characteristics of speakers for inter-session variability compensation. Universal background model (UBM) [114], i-vector extractor [28], nuisance attribute projection (NAP) matrix [19], probabilistic linear discriminant analysis (PLDA) model [79] are well known background models.

2.2 Datasets used in an ASVS

Three kinds of speech datasets are used in an ASVS.

1. **Enrollment Dataset,** $\mathcal{E}$: It contains speech files from speakers who would like to be enrolled in the system during the training phase of the ASVS. Enrolled speakers are the target speakers in the authentication phase.
2. **Authentication Dataset,** $\mathcal{A}$: It contains speech files from speakers who are verified/ authenticated by the ASVS in the authentication phase. This dataset contains speech files both from clients and impostors.

3. **Background Dataset,** $\mathcal{B}$: It contains multi-channel utterances from several hundred speakers, resulting in several thousands of speech files which are collected from multiple databases. This dataset is used for training background models and it does not contain any data included in $\mathcal{A}$.

The combination of $\mathcal{E}$ and $\mathcal{A}$ used for measuring performance of an ASVS is known as an evaluation set, $\mathcal{V}$. When an evaluation set is used for tuning parameters and deciding threshold value it is called a development set, $\mathcal{D}$.

2.2.1 The NIST-SRE Datasets

From 1996 to 2006, the NIST organized an speaker recognition evaluation (SRE) campaign annually. After that this campaign has been being organized once in two years. These campaigns are mainly funded by the U.S. Department of Defense. In each campaign, a large, specifically designed speech corpora having mainly conversational speech is distributed to the participants without any charge as an evaluation dataset. Now these days, researchers in the text-independent speaker verification field mainly use NIST-SRE datasets as evaluation datasets. Among the NIST-SRE datasets, NIST SRE 2012 dataset for core task is the largest one. In the NIST SRE framework, except the NIST SRE 2012, $\mathcal{E}$ is not allowed to be included in $\mathcal{B}$.

2.2.2 Switchboard Datasets

Switchboard datasets are popular as background datasets to the researchers in the text-independent ASVS. The linguistic data consortium (LDC) is in charge of maintenance and distribution of Switchboard datasets. The LDC, formed in 1992, is an open consortium of universities, libraries, corporations and government research laboratories. For LDC members for a specific year, Switchboard databases are free for that year, for non-members it costs money. Switchboard II Phase-1,-2,-3, Switchboard Cellular Part-1 and Part-2 are popular as $\mathcal{B}$.

2.3 Feature Extraction

The feature extraction module of a speaker verification system transforms the raw speech signal into feature vectors. One reason for feature extraction is to emphasize the speaker specific characteristics. As shown in Figure 2.2, in the raw speech signal the inter-speaker variability is not easily noticeable for all speakers. Figure 2.3 reveals that in mel-frequency cepstral coefficient (MFCC) feature, high inter-speaker variability is easily noticeable. For example, for Speaker-1, the 3rd MFCC component is always higher than the 2nd in the later region of the utterances but for the other two speakers there no such pattern. Another reason of feature extraction is to reduce the dimension of the speech data. For example, 15 dimensional 90 feature vectors are more desirable than the 1100 dimensional one speech signal because of:

- The joint distribution for high-dimensional features are hard to estimate. Accordingly, reliable speaker modeling using such features requires large amounts of data.
- Low-dimensional features need low computational cost.

2.3.1 Properties of an Ideal Feature

The feature extraction method is very important for an automatic speaker verification system because the quality of the speaker modeling and pattern matching is strongly determined by the quality of the extracted features. An ideal feature would [135]:

- have large inter-speaker variability (i.e., between-speaker variability) and small intra-speaker variability (i.e., within-speaker variability)
- be robust against noise
- occur frequently and naturally in speech
- be easy to measure from speech signal
- be difficult to impersonate/mimic
- not be affected by speaker's health or long-term variation in voice

2.3.2 Types of Features

There are different ways to categorize the features. From the viewpoint of their physical interpretation, we can divide the speech features into the

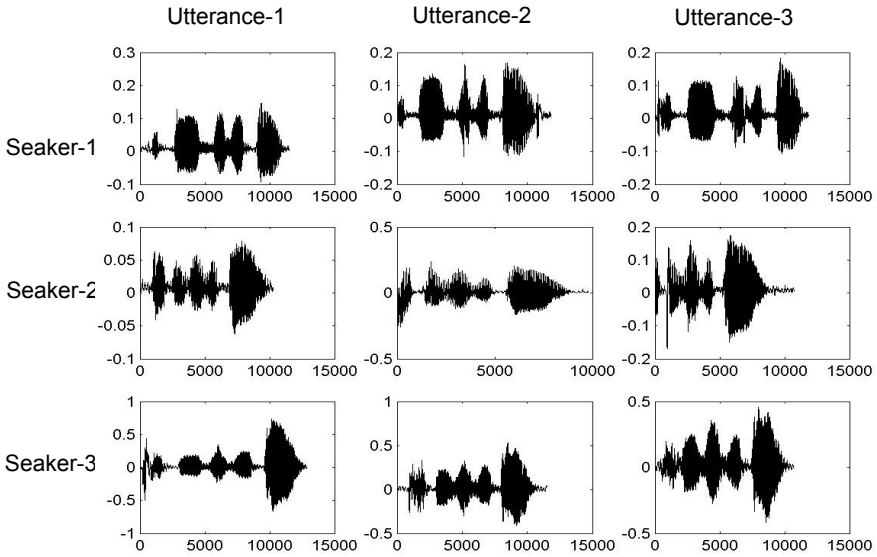

Figure 2.2: Raw speech signal of sample speech (the Bengali word "Protijogeeta", meaning "competition" in English) segments from three speakers. Inter-speaker variability is not noticeable for the utterances of *Speaker-2* and *Speaker-3*. Utterance-2 of Speaker-2 seems to be similar to Utterance-1 and Utterance-2 of Speaker-3; where Utterance-1 and Utterance-2 of Speaker-1 and Utterance-3 of Speaker-3.

following five categories [85]:

1. **Short-term Spectral Feature**: This type of feature are computed from short frames of about 20-30 ms in duration. They are usually descriptors of the short-term *spectral envelope* which is an acoustic correlate of *timbre*, i.e., the “color" of sound, as well as the resonance properties of the supralaryngeal vocal tract.

2. **Voice-source Feature**: This type of features characterize the glottal excitation signal of voiced sounds such as glottal shape and fundamental frequency.

3. **Spectro-temporal feature**: This type of features describes formant transitions and energy modulations which contain speaker-specific information.

4. **Prosodic Feature**: This type of features refers to non-segmental aspects of speech such as syllable stress, intonation patterns, speaking

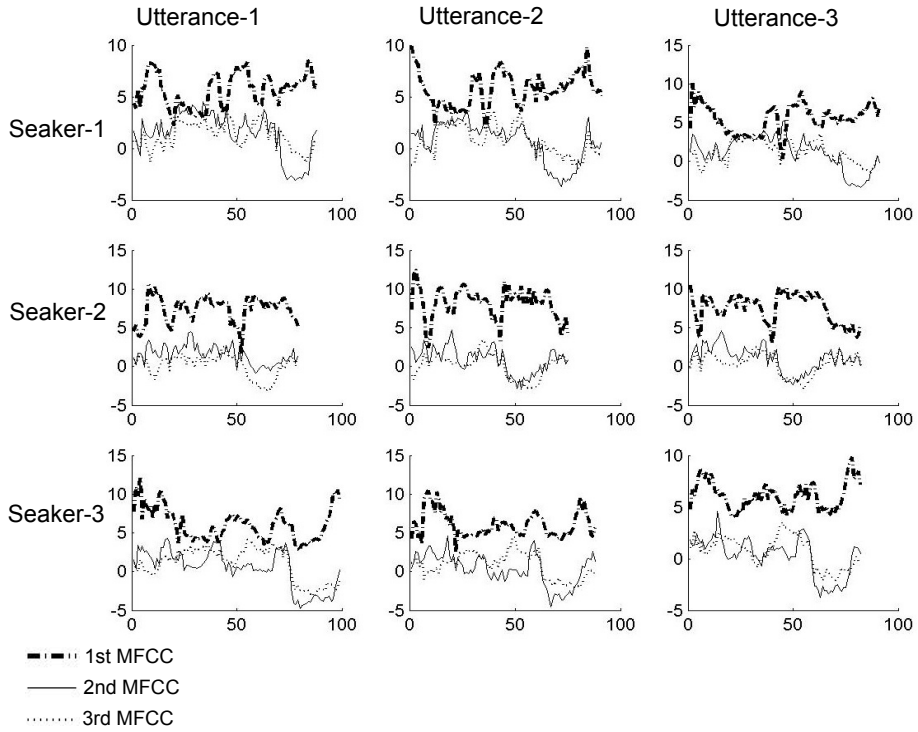

Figure 2.3: 1st, 2nd and 3rd MFCCs of the utterances whose waveforms are depicted in Figure 2.2. Notice MFCCs have high inter-speaker variability and low intra-speaker variability.

rate, rhythm etc.

5. **High-level Feature**: This type of features captures conversation-level characteristics of speakers, such as characteristic use of words(“uh-hh", “right", “you know",“ah! so",“I see",“sokka sokka",“for example" etc.)

Any features is not globally accepted as the best feature. The choice of a feature depends on the intended application, computing resources, amount of speech data available and whether the speakers are co-operative or not. A comparison of the five types of features are given in Figure 2.4. Since, high-level features are easier to impersonate, so in speaker verification it is better to use short-term spectral features. Two popular short-term spectral features are Mel-frequency Cepstral Coefficient (MFCC) and Perceptual

Linear Prediction (PLP) coefficients.

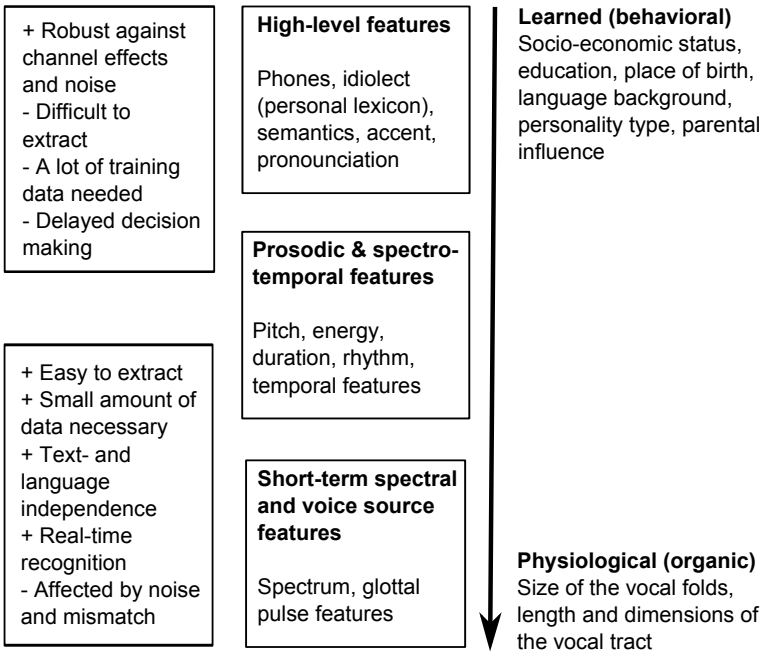

Figure 2.4: Comparison of different types of features. The leftmost boxes include positive and negative sides of features, middle boxes include different kinds of features with examples, and the factors affect the features are pointed out in the rightmost part.

2.3.3 Pre-processing

Before extracting features, a speech signal is passed through the following pre-processing steps:

1. **Silence Removing**: At this step, speech endpoint detection algorithm is used to detect the presence of speech, and to remove pauses and silences. It is an important step because of the following two counts:

   (a) Firstly, about half of the input speech signal contains silence. The periods of silence contain no acoustical contents, and provide no information to aid the speaker verification process. As a result, storing these periods of silence is nothing important rather wastage of resources.

(b) Secondly, if the silence portions are left in the speech signal, they will form part of the features and affect the identification accuracy.

For telephone speech, it is considered quite noiseless signal, so short-term energy or average magnitude based method is enough to remove the silence part of the speech segment.

2. **Pre-emphasizing**: Pre-emphasis refers to filtering that emphasizes the higher frequencies. Its purpose is to balance the spectrum of voiced sounds that have a steep roll-off in the high frequency region. It is generally done by the following way:

$$\hat{s}_n = s_n - \alpha \times s_{(n-1)} \tag{2.2}$$

where α is the pre-emphasis co-efficient which should be in the range $0 \leq \alpha \leq 1$. Generally 0.97 is used for α.

3. **Framing**: Since the speech signal changes continuously due to the articulatory movements of the vocal production organs, the signal must be processed in short frames, within which the parameters remain quasi-stationary. Therefore, in this step, a continuous speech signal is blocked into frames of N samples, with adjacent frames being separated by $M(M < N)$ samples. The overlap between adjacent frames is used to smooth the frame-to-frame transitions and to provide a better handling of the correlation existing between successive parts of the voice signal. Typically a frame length of 10-30 milliseconds is used. A typical frame overlap is around 30 to 50% of the frame size.

4. **Windowing**: The purpose of the windowing is to reduce the effect of the spectral artifacts that result from the framing process. The most commonly used window function in speech processing is the Hamming window defined as follows:

$$w(n) = 0.54 - 0.46\cos(\frac{2\pi(n-1)}{N-1}) \tag{2.3}$$

where N represents the width of window function in samples. Typically, it is an integer power-of-2, such as $2^{10} = 1024$. n is an integer with values $0 \leq n \leq N$.

2.3.4 Perceptual Linear Prediction

In perceptual linear prediction (PLP) technique proposed by Hynek Hermanskey, auditory like spectrum of speech is approximated. The following steps of PLP extraction closely follow the paper written by Hermanskey [64]:

1. The windowed speech segment is transformed into the frequency domain by discrete Fourier transform (DFT). The short-term power spectrum of the short term speech spectrum is:

$$P(\omega) = Re[S(\omega)]^2 + Im[S(\omega)]^2, \tag{2.4}$$

   where Re and Im mean $Real$ and $Imaginary$. $S(\omega)$ means short-term speech spectrum.
2. The spectrum $P(\omega)$ is warped along its frequency axis ω into the Bark frequency Ω by

$$\Omega(\omega) = 6\ln\{\omega/1200\pi + [(\omega/1200\pi)^2 + 1]^{0.5}\}, \tag{2.5}$$

   where ω is the angular frequency in rad/s.
3. The critical-band power spectrum is generated from the warped $P(\Omega)$ as follows:

$$\Theta(\Omega_i) = \sum_{\Omega=-1.3}^{2.5} P(\Omega - \Omega_i)\Psi(\Omega), \tag{2.6}$$

   where $\Psi(\Omega)$ is the critical band curve which is given by:

$$\Psi(\Omega) = \begin{cases} 0 & \text{for} \quad \Omega < -1.3, \\ 10^{2.5(\Omega+0.5)} & \text{for} \quad -1.3 \leq \Omega \leq < -0.5, \\ 1 & \text{for} \quad -0.5 < \Omega < 0.5, \\ 10^{-1.0(\Omega-0.5)} & \text{for} \quad 0.5 \leq \Omega \leq 2.5, \\ 0 & \text{for} \quad \Omega > 2.5, \end{cases} \tag{2.7}$$

4. $\Theta[\Omega(\omega)]$ is sampled in approximately 1-Bark intervals.
5. The sampled $\Theta[\Omega(\omega)]$ is pre-emphasized by simulated equal-loudness curve

$$\Xi[\Omega(\omega)] = E(\omega)\Theta[\Omega(\omega)], \tag{2.8}$$

   where $E(\omega)$ is an approximation to the nonequal sensitivity of human hearing at different frequencies which is given by

$$E(\omega) = \frac{(\omega^2 + 56.8 \times 10^6)\omega^4}{(\omega^2 + 6.3 \times 10^6) \times (\omega^2 + 0.38 \times 10^9)}. \tag{2.9}$$

6. The values of the first and last samples are made equal to the values of their nearest neighbors so that $\Xi[\Omega(\omega)]$ begins and ends with two equal-valued samples.
7. $\Xi[\Omega(\omega)]$ is compressed as follows:

$$\Phi(\Omega) = \Xi(\Omega)^{0.33} \tag{2.10}$$

8. The inverse DFT (IDFT) is applied to $\Phi(\Omega)$ to yield the auto correlation function dual to $\Phi(\Omega)$.
9. The first $M + 1$ auto correlation values are used to get autoregressive coefficients of the M-th order all-pole model.
10. The autoregressive coefficients are transformed into cepstral coefficients.

2.3.5 Mel-Frequency Cepstral Coefficient

MFCC was introduced by Davis and Mermelstein [24] in speech and audio processing which was then become popular features in speaker recognition field. There are many similarities between the processes of MFCC and PLP extraction, which are mentioned below:

1. **Spectral Analysis**: Like PLP, at first short-term power spectrums are obtained by applying DFT to the windowed frames for MFCC .

2. **Critical-band Analysis**: A frequency band is a set of consecutive frequencies and a critical band is a frequency band within which frequencies cannot be distinguished from the center frequency by the human auditory system. Like PLP, MFCC also does critical-band analysis and employs an auditory-based frequency warping of the frequency axis derived from the frequency sensitivity of human hearing. The small difference is that MFCC is based on a uniform spacing along the Mel-scale and PLP uses the Bark-scale. The Mel-scale is defined by:

$$f_{mel} = 2595\log_{10}(1 + \frac{f}{700}) \tag{2.11}$$

   In mel-scale, the size of critical-band is approximated by:

$$BW_{critical}(mel) = 25 + 75[1 + 1.4(\frac{f_{mel}}{1000})^2]^{0.69} \tag{2.12}$$

Whereas the Bark-scale is defined by

$$f_{bark} = 13\arctan(0.00076f) + 3.5\arctan(\frac{f}{7500})^2 \tag{2.13}$$

In bark-scale, the size of critical-band is approximated by:

$$BW_{critical}(bark) = \frac{52548}{(f_{bark})^2 - 52.56f_{bark} + 69039} \tag{2.14}$$

3. **Homomorphic Analysis**: In MFCC the log filterbank amplitude is taken whereas in PLP cubic-root amplitude is done.

4. **Cepstral Analysis**: In MFCC, cepstral coefficients are calculated from the $\log$ Mel filter using a discrete cosine transform(DCT). In PLP, as described in Section 2.3.4 cepstral co-efficients are computed from LP co-efficients.

In short, MFCC can be calculated as follows:

$$c_i = \sqrt{\frac{2}{L}}\sum_{l=1}^{L}[\log Y_l]\cos[\frac{\pi n}{L}(l-0.5)] \tag{2.15}$$

where L is the number of mel-filterbank channels and Y_l is the log filterbank amplitudes.

2.3.6 Dynamic Features

The features described so far are known as static features. Static features capture the average frequency distribution during a frame. Furui [41], [42] showed that adding time derivatives to static features can greatly enhance the performance of speaker verification systems. The first-order dynamic feature or Δ co-efficient at time t can be computed using the following regression formula [138]:

$$\Delta_t = \frac{\sum_{\theta=1}^{\Theta}\theta(c_{t+\theta} - c_{t-\theta})}{2\sum_{\theta=1}^{\Theta}\theta^2} \tag{2.16}$$

$c_{t\pm\theta}$ is the static feature. The value of Θ is generally set to 1. The end-effect problem for the first and last frame can be solved:

$$\Delta_t = c_{t+1} - c_t, \quad t < \Theta \tag{2.17}$$

and

$$\Delta_t = c_t - c_{t-1}, \quad t \geq T - \Theta \tag{2.18}$$

where T is the number of total frames. The second-order dynamic feature or $\Delta\Delta$ co-efficient can be computed in the same way using Δ_t in place of static feature.

2.4 Speaker Modeling

Modeling speaker characteristics is an important part in an ASVS. In the training phase of an ASVS, speaker models are built up which are used to generate scores in the authentication phase. In 2000 Reynolds et al. [114] introduced Gaussian mixture models (GMM)s in ASVS for modeling speaker characteristics. Since then, most works in text-independent ASVS have been done either by using the standard GMM system [114] or by extending the concept of GMM. Before 2005, most popular GMM based systems were GMM-UBM which used speaker-specific GMM for modeling the target speaker and speaker-independent GMM known as UBM for modeling impostors during score generation. In 2005 Kenny et al. [78] proposed to use stacked GMM mean vectors called *supervectors* to model speaker and inter-session variability separately using factor analysis. This kind of ASVS is known as JFA based ASVS. In 2006 Campbell et al. [18] showed that a support vector machine (SVM) using GMM-supervector (GSV) based linear kernel outperforms the standard GMM-UBM configuration. This kind of ASVS is quite often called GMM-SVM system with GSV linear kernel. In 2009 Dehak et al. [26] proposed to map high-dimensional GSVs to low-dimensional vectors by using factor analysis. This kind of systems are known as *i-vectors* based systems. The common part of all these models is GMM which is discussed in Section 2.4.1. An SVM based system is discussed in Section 2.4.2. GMM-SVM with GSV linear kernel and i-vector based models are discussed in Section 2.4.3 and Section 2.4.4, respectively. JFA based ASVS is not discussed here since we do not use this model in this thesis.

2.4.1 Gaussian Mixture Model (GMM)

A GMM is a probabilistic model for estimating density of a random variable by combining a set of *Gaussian* or *normal* distributions. Each member of the

distribution set is called a Gaussian component. The equation of the density of a GMM with M Gaussian components can be written as:

$$p(\boldsymbol{x} \mid \lambda) = \sum_{m=1}^{M} w_m \mathcal{N}_m(\boldsymbol{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m), \tag{2.19}$$

where $\boldsymbol{x}$ is a *D*-dimensional random vector; $\mathcal{N}_m(.)$ is the *i*-th Gaussian component density with mean $\boldsymbol{\mu}_m$ and co-variance matrix $\boldsymbol{\Sigma}_m$; and w_m is the mixture weight. Each component density is a D-variate Gaussian function of the form:

$$\mathcal{N}_m(\boldsymbol{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_m|^{1/2}} exp\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_m)'\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_m)\}. \tag{2.20}$$

The mixture weights are the prior probability of M Gaussian components which satisfy the following constraint:

$$\sum_{m=1}^{M} w_m = 1. \tag{2.21}$$

The complete set of parameters of a GMM are represented by the following notation

$$\lambda = \{w_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\} \qquad m = 1, 2, ...M \tag{2.22}$$

Maximum Likelihood Parameter Estimation

The most popular and well-established method to estimate λ of a GMM is the maximum likelihood (ML) estimation. In this method, the model parameters which maximize the likelihood of the GMM, given the training data, X, are selected as the model parameters:

$$\hat{\lambda} = \arg\max_{\lambda} p(X \mid \lambda) \tag{2.23}$$

where X is a sequence of T training vectors i.e., $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T, \}$ and the likelihood function $p(X \mid \lambda)$ can be written as:

$$p(X \mid \lambda) = \prod_{t=1}^{T} p(\boldsymbol{x_t} \mid \lambda). \tag{2.24}$$

By taking $\log$ in both sides of Equation (2.24), the likelihood function can be written as:

$$L(X \mid \lambda) = \sum_{t=1}^{T} \log(p(\boldsymbol{x_t} \mid \lambda)). \tag{2.25}$$

where

$$L(X \mid \lambda) = \log(p(X \mid \lambda)) \tag{2.26}$$

Using Equation (2.19), Equation (2.25) can be written as:

$$L(X \mid \lambda) = \sum_{t=1}^{T} \log(\sum_{m=1}^{M} w_m \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)). \tag{2.27}$$

Unfortunately Equation (2.27) is difficult to calculate. The numerical difficulties come from:

- the sum inside the $\log$ and
- $\mathcal{N}(.)$ is a nonlinear function of the parameters $\boldsymbol{\mu}_m$ and $\boldsymbol{\Sigma}_m$.

Therefore, direct maximization of the *log-likelihood* is not possible. However, ML parameters can be estimated by using the expectation-maximization (EM) algorithm. The basic idea of this algorithm is to start with an initial model parameter set λ and estimate a new model parameter set $\bar{\lambda}$ such that $p(X \mid \bar{\lambda}) \geq p(X \mid \lambda)$. The new model parameter set then becomes the initial model parameter set for the next iteration and the process is repeated until some convergence threshold is reached. The steps of EM algorithm are given below [7]:

1. Initialize the means $\boldsymbol{\mu}_m$, covariances $\boldsymbol{\Sigma}_m$ and mixing coefficients or weights w_m, and evaluate the initial value of the log-likelihood.

2. **E step.** Compute the posterior probabilities using the current parameter values
$$\gamma_{mt} = \frac{w_m \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}{\sum_{i=1}^{M} w_i \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \tag{2.28}$$

3. **M step.** Re-estimate the parameters using the current posterior probabilities
$$\boldsymbol{\mu}_m^{new} = \frac{1}{N_m} \sum_{t=1}^{T} \gamma_{mt} \boldsymbol{x}_t, \tag{2.29}$$
$$\boldsymbol{\Sigma}_m^{new} = \frac{1}{N_m} \sum_{t=1}^{T} \gamma_{mt} (\boldsymbol{x}_t - \boldsymbol{\mu}_m^{new})(\boldsymbol{x}_t - \boldsymbol{\mu}_m^{new})' \tag{2.30}$$
$$w_m^{new} = \frac{N_m}{N} \tag{2.31}$$
where
$$N_m = \sum_{t=1}^{T} \gamma_{mt} \tag{2.32}$$

4. Evaluate the log-likelihood using Equation (2.27) and check for the convergence of either the parameters or the log-likelihood. If the convergence criterion is not satisfied return to Step 2.

In an GMM based ASVS, feature vectors extracted from speech segments are considered as the random variable $\boldsymbol{x}$. When a GMM is trained by feature vectors extracted from thousands speech files from hundreds speakers in order to represent speaker-independent distributions, that GMM is called UBM. When a GMM is trained to represent speaker-specific distribution, then that GMM is called a speaker-specific GMM. Generally a speaker-specific GMM is trained by adapting an UBM using feature vectors extracted from the speaker (See Section 2.5 for various adaptation techniques).

2.4.2 Support Vector Machine (SVM)

In an SVM [128] an optimal hyperplane $\mathcal{H}$ is found out which provides the maximum *margin* between a positive class and a negative class. For n number of data points of two classes, $\{\boldsymbol{x}_i, y_i\}_{i=1}^{n}$, where $y_i \in \{+1, -1\}$ is the class label of the data point $\boldsymbol{x}_i$, a linear hyperplane $\mathcal{H}$ can be written as:

$$\mathcal{H} : \boldsymbol{w}' \cdot \boldsymbol{x} + b = 0 \tag{2.33}$$

where $\boldsymbol{w}$ is perpendicular to $\mathcal{H}$ and $b \in \mathbb{R}$ is a constant. $\mathcal{H}$ should be as far away from the data points of both classes as possible. Let d_+ and d_- be the shortest distance from $\mathcal{H}$ to the closest positive class data point and to the closest negative class data point, respectively, and $d_+ = d_-$. Then, the margin, $\mathcal{M}$, can be written as:

$$\begin{aligned} \mathcal{M} &= d_+ + d_- \\ &= \frac{1}{||\boldsymbol{w}||} + \frac{1}{||\boldsymbol{w}||} \\ &= \frac{2}{||\boldsymbol{w}||} \end{aligned} \tag{2.34}$$

where $||\boldsymbol{w}||$ is the Euclidean norm of $\boldsymbol{w}$. In order to find the maximum $\mathcal{M}$, we need to find a pair of hyperplanes, $\mathcal{H}_1$ and $\mathcal{H}_2$, which minimize $||\boldsymbol{w}||^2$. Note that $\mathcal{H}_1$ and $\mathcal{H}_2$ are parallel of $\mathcal{H}$ and can be written as:

$$\mathcal{H}_1 : \boldsymbol{w}' \cdot \boldsymbol{x} + b = 1 \tag{2.35}$$

$$\mathcal{H}_2 : \boldsymbol{w}' \cdot \boldsymbol{x} + b = -1 \tag{2.36}$$

Ideally, there should be nothing between $\mathcal{H}_1$ and $\mathcal{H}_2$ except $\mathcal{H}$. However, it is rare to have data points of two classes which are completely separable. Therefore, there can be some data points in the middle of $\mathcal{H}_1$ and $\mathcal{H}_2$. These inseparable data points can be handled by positive slack variables. Therefore, the optimization problem of $\mathcal{H}$ can be summarized as:

$$\text{Minimize } \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n}\xi_i \tag{2.37}$$

$$\text{subject to } y_i(\boldsymbol{w}' \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

where ξ_i is the slack variable in the optimization, $\xi_i = 0$ when no error occurs to separate x_i, and C is the tradeoff parameter between error and margin which can be adjusted by the user. The above optimization problem can be easily solved by turning it into a convex quadratic programming (QP) problem as follows:

$$\text{Maximize } \theta(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i' \cdot \boldsymbol{x}_j \tag{2.38}$$

$$\text{subject to } C \geq \alpha_r \geq 0 \text{ and } \sum_{i=1}^{n}\alpha_i y_i = 0$$

where $r \in \{i, j\}$, α_r is a Lagrange multiplier, C is the tradeoff parameter between error and margin, and generally, C is chosen by cross validation. In the training phase of an SVM-based system, the main target is to find the data points which lie on $\mathcal{H}_1$ and $\mathcal{H}_2$. These data points are called *support vectors*, since their removal can change the solution.

In reality, linear hyperplane cannot separate low dimensional data points of two classes even after introducing slack variables. On the other hand, non-linear hyperplane is complicated to be found out. This situation can be avoided by transforming data points from their original space called *input space* or *data space* into a high dimensional (possibly infinite dimensional) space called *feature space*, where linear hyperplane separator can be used. The above convex QP problem in 2.38 can be written as:

$$\text{Maximize } \theta(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \Phi(\boldsymbol{x}_i)' \cdot \Phi(\boldsymbol{x}_j) \tag{2.39}$$

$$\text{subject to } C \geq \alpha_r \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0$$

Doing dot product in a very high dimensional space is costly, which can be avoided by using a kernel function, $K(.,.)$. A kernel function can estimate the dot product in the input space which is equivalent to doing dot product in the feature space. Using a kernel function, the above convex QP problem in 2.39 can be written as:

$$\text{Maximize } \theta(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{2.40}$$

$$\text{subject to } C \geq \alpha_r \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0$$

where,

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)' \cdot \Phi(\boldsymbol{x}_j). \tag{2.41}$$

In an SVM based ASVS, the data points from a target speaker are considered in *Class +1*, whereas the data points from impostors are considered in *Class -1*. Different types of kernel functions have been proposed to convert variable length sequence of feature vectors into fixed-dimension vectors for scoring by SVM. Some examples of kernels are the generalized linear discriminant sequence (GLDS) kernel [16], Fisher kernel [131], n-gram kernel[17], GMM-supervector (GSV) linear kernel [18, 19], GSV non-linear kernel [25], linearized LR kernel [141] etc. In this thesis we use GSV linear kernel which is discussed in details in the next section.

2.4.3 GMM-SVM Modeling

In text-independent speaker verification, a GMM-SVM based ASVS was proposed to associate robustness of the GMM system with discriminative power of the SVM system [91, 131, 18, 19, 25]. One of the popular GMM-SVM systems uses GSV linear kernel written as:

$$K(a, b) = \sum_{m=1}^{M} (\sqrt{w_m} \boldsymbol{\Sigma}_m^{-\frac{1}{2}} \boldsymbol{\mu}_m^{(a)})' (\sqrt{w_m} \boldsymbol{\Sigma}_m^{-\frac{1}{2}} \boldsymbol{\mu}_m^{(b)}) \tag{2.42}$$

This kernel was proposed by Cambell et al. in [18, 19]. In this system a very high-dimensional vector is built by concatenating the mean vectors of all Gaussian components of the speaker-specific GMM. This high-dimensional

vector is called GMM supervector (GSV). In the training phase, a linear kernel is used and a set of positive and negative supportvectors are found out for each target speakers. In the authentication phase support vectors are used to generate scores for authentication speech segments. Steps of these two phased are described in details below:

- **Training Phase**:
  1. Extract D-dimensional feature vectors (e.g., MFCC, PLP co-efficients etc) from all utterances of enrollment dataset, $\mathcal{E}$, and background datasets, $\mathcal{B}_1$ and $\mathcal{B}_2$. Most of the cases $\mathcal{B}_2 \subset \mathcal{B}_1$.
  2. Train a UBM, λ, with M Gaussian components applying the EM algorithm described in Section 2.4.1 on the feature vectors extracted from $\mathcal{B}_1$, where each Gaussian component of λ can be written as:

$$\lambda_m = \{w_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\},\ m = 1, 2, ..., M, \tag{2.43}$$

     where $w_m \in \mathbb{R}^1$ is the mixture weight, $\boldsymbol{\mu}_m \in \mathbb{R}^{D\times 1}$ and $\boldsymbol{\Sigma}_m \in \mathbb{R}^{D\times D}$ are the mean vector and the co-variance matrix of m-th Gaussian components, respectively.
  3. Train either only one speaker-specific GMM, $\lambda^{(S)}$, or P number of speaker-specific GMMs, $\lambda^{(S1)}, \lambda^{(S2)}, ..., \lambda^{(SP)}$, for each target speaker, $S \in \mathcal{E}$, by adapting the UBM with the feature vectors extracted from $P \geq 1$ speech segments of $S \in \mathcal{E}$ following any adaptation technique described in Section 2.5.
  4. Train speaker-specific GMMs, $\lambda^{(I1)}, \lambda^{(I2)}, ..., \lambda^{(IQ)}$, for Q number of background speakers where $I \in \mathcal{B}_2$.
  5. Make a supervector for each GMM by concatenating the mean vectors of all Gaussian components. If the mean vector of each Gaussian component is D-dimensional then the supervector of a GMM with M Gaussian components will be $MD \times 1$-dimensional which can be written as:

$$\boldsymbol{\mu}^{(n)} = \{\sqrt{w_m}\boldsymbol{\Sigma}_m^{-\frac{1}{2}}\boldsymbol{\mu}_1^{(n)}, \sqrt{w_m}\boldsymbol{\Sigma}_m^{-\frac{1}{2}}\boldsymbol{\mu}_2^{(n)}, ..., \sqrt{w_m}\boldsymbol{\Sigma}_m^{-\frac{1}{2}}\boldsymbol{\mu}_M^{(n)}\}, \tag{2.44}$$

     where $n \in \{S, I\}$, $S \in \mathcal{E}$ and $I \in \mathcal{B}_2$.
  6. Prepare a training dataset for each $S \in \mathcal{E}$ by putting P supervectors from S and Q supervectors from $I \in \mathcal{B}_2$ with class labels $+1$

and -1, respectively. The training data set can be written as:

$$\{\boldsymbol{\mu}^{(S1)}, +1\}, ..., \{\boldsymbol{\mu}^{(SP)}, +1\}, \{\boldsymbol{\mu}^{(I1)}, -1\}, ..., \{\boldsymbol{\mu}^{(IQ)}, -1\}$$

7. Train a speaker-specific SVM for each $S \in \mathcal{E}$.
   (a) Solve the following convex quadratic programming problem equivalent to Problem 2.37 using the training dataset.

$$\begin{aligned} \text{Maximize } \theta(\alpha) &= \sum_{i=1}^{P+Q} \alpha_i - \frac{1}{2} \sum_{i=1}^{P+Q} \sum_{j=1}^{P+Q} \alpha_i \alpha_j y_i y_j \boldsymbol{\mu}^{(i)} \boldsymbol{\mu}^{(j)} \\ \text{subject to } C \geq \alpha_r \geq 0 \text{ and } &\sum_{i=1}^{P+Q} \alpha_i y_i = 0 \end{aligned} \tag{2.45}$$

   where $r \in \{i, j\}$, α_r is a Lagrange multiplier, y_r is the class label of r-th supervector, C is the tradeoff parameter between error and margin. Generally, C is chosen by cross validation.

$$y_r = \begin{cases} +1 & \text{for target speaker's supervectors,} \\ -1 & \text{for background supervectors} \end{cases}$$

   (b) Pick the supervectors for which $\alpha_i > 0$ as supportvectors, $\boldsymbol{v}$.
   (c) Estimate $\boldsymbol{w}$ by using all $\boldsymbol{v}$.

$$\boldsymbol{w} = \sum_{g=1}^{G} \alpha_i y_i \boldsymbol{v}_i \tag{2.46}$$

   where G is the total number of positive and negative supportvectors.
   (d) Using a single supportvector either on $\mathcal{H}_1$ or on $\mathcal{H}_2$, estimate the value of b. For example, using a positive supportvector, $\boldsymbol{v}_+$ and $\boldsymbol{w}$ from Eq. 2.46, b can be estimated from Eq. 2.35 as:

$$b = \boldsymbol{w}' \cdot \boldsymbol{v}_+ - 1 \tag{2.47}$$

Figure 2.5 is a schematic example of training a GMM-SVM based ASVS using one speech segment from a target speaker and Q speech segments from Q background speakers.

- **Authentication Phase**:

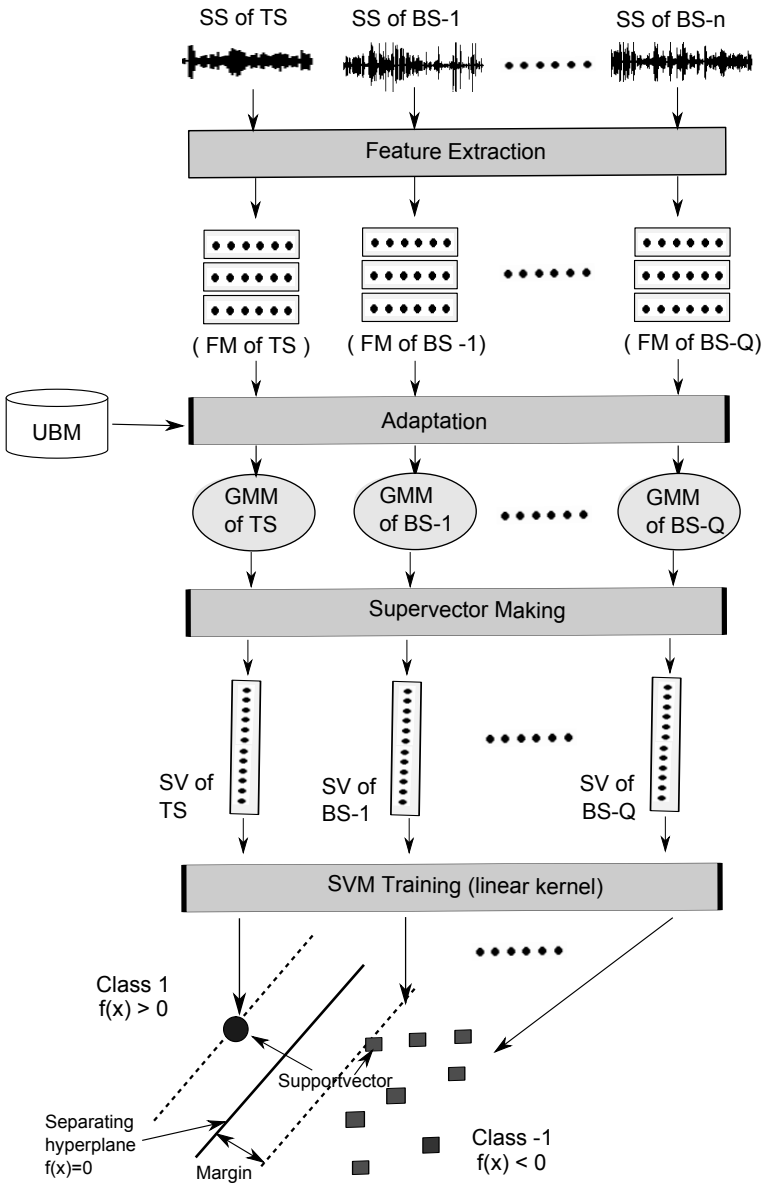

Figure 2.5: GMM-SVM Training. The meaning of notations are– SS: Speech Segment, BS: Background Speaker, TS: Target Speaker, FM: Feature Matrix, SV: Supervector, UBM: Universal Background Model.

1. Extract feature vectors from the authentication speech segment, x.
2. Train GMM, $\lambda^{(x)}$, by adapting UBM using feature vectors of x.
3. Make a supervector, $\boldsymbol{\mu}^{(x)}$, by concatenating means of $\lambda^{(x)}$.

4. Solve the following equation to get score of x against the target speaker S's SVM:

$$Score(S, x) = \sum_{g=1}^{G} \alpha_g y_g \boldsymbol{\mu}^{(x)} \boldsymbol{\upsilon}_g^{(S)} + b. \tag{2.48}$$

5. Take decision about x by comparing $Score(S, x)$ with a predeclared threshold.

2.4.4 i-Vector based Modeling

An i-vector based system [26, 28] assumes that the feature vectors of an utterance are drawn independently from a GMM. The stacked mean vectors of the GMM constitute a speaker- and channel-dependent *GMM-supervector*, $\boldsymbol{\mu}$. It is assumed that $\boldsymbol{\mu}$ is generated according to

$$\boldsymbol{\mu} = \bar{\boldsymbol{\mu}} + \mathbf{T}\boldsymbol{\phi}, \tag{2.49}$$

where $\bar{\boldsymbol{\mu}}$ is the mean of speaker- and channel-independent supervectors, $\mathbf{T}$ is a basis for the *total variability subspace*, and $\boldsymbol{\phi}$ is a random vector. It is assumed that $\boldsymbol{\phi}$ follows the standard normal distribution and its dimension is lower than that of $\bar{\boldsymbol{\mu}}$.

Given the features from an utterance, the i-vector, $\boldsymbol{\omega}$, is the *maximum a posteriori* (MAP) estimate of $\boldsymbol{\phi}$. The mathematical framework for training $\mathbf{T}$ and estimating $\boldsymbol{\phi}$ is the same as used for training the eigenvoice matrix, $\mathbf{V}$, and estimating the hidden variable, $\mathbf{y}$, in the eigenvoice MAP [82]. The only difference is that, in the eigenvoice MAP, $\mathbf{y}$ is the same for all utterances of the same speaker, whereas in an i-vector based system, $\boldsymbol{\phi}$ is different from utterance to utterance. The steps for building i-vector based ASVS are given below:

- **Training Phase**:
  1. Extract D dimensional feature vectors from all utterances of background set, $\mathcal{B}_1$ and $\mathcal{B}_2$. The same datasets can be used as $\mathcal{B}_1$ and $\mathcal{B}_2$.
  2. Train a UBM using feature vectors extracted from $\mathcal{B}_1$.
  3. Train an i-vector extractor, i.e., $\mathbf{T}$-matrix using feature vectors extracted from $\mathcal{B}_2$:

(a) For each utterance u with T feature vectors $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$, estimate the 0th and the centralized 1st order Baum-Welch statistics, N_m and F_m, respectively, using the m-th Gaussian component of UBM as follows:

$$\begin{aligned} N_m(u) &= \sum_{t=1}^{T} \gamma_{mt}(u) \\ &= \sum_{t=1}^{T} \frac{w_m \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}{\sum_{i=1}^{M} w_i \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \end{aligned} \tag{2.50}$$

$$F_m(u) = \sum_{t=1}^{T} \gamma_{mt}(u)(\boldsymbol{x}_t - \boldsymbol{\mu}_m), \tag{2.51}$$

(b) Decide a desired rank R and initialize the i-vector extractor $\mathbf{T} \in \mathbb{R}^{MD \times R}$ randomly.

(c) Estimate precision matrix, $\mathbf{L}(u)$as follows:

$$\mathbf{L}(u) = \mathbf{I} + \sum_{m=1}^{M} N_m(u) \mathbf{T}'_m \boldsymbol{\Sigma}_m^{-1} \mathbf{T}_m \tag{2.52}$$

where $\mathbf{T}_m \in \mathbb{R}^{D \times R}$ is the m-th sub-matrix of $\mathbf{T}$.

(d) Estimate i-vector of utterance u as follows:

$$\boldsymbol{\omega}(u) = \mathbf{L}^{-1}(u) \sum_{m=1}^{M} \mathbf{T}'_m \boldsymbol{\Sigma}_m^{-1} F_m(u) \tag{2.53}$$

(e) Estimate accumulators, **C** and **A** as follows:

$$\mathbf{C} = \sum_{u \in \mathcal{B}_2} F(u) \boldsymbol{\omega}(u)^{'} \tag{2.54}$$

$$\mathbf{A}_m = \sum_{u \in \mathcal{B}_2} N_m(u) (\mathbf{L}^{-1}(u) + \boldsymbol{\omega}(u) \boldsymbol{\omega}(u)^{'}) \tag{2.55}$$

where $F(u) = (F_1(u)', F_2(u)', ..., F_M(u)')'$

(f) Update $\mathbf{T}_m$ as follows:

$$\mathbf{T}_m = \mathbf{C}\mathbf{A}_m^{-1} \tag{2.56}$$

(g) Iterate Step 3c-Step 3f until $\mathbf{T}$ converges.

- **Authentication Phase**:

1. Estimate the centralized 1st order Baum-Welch statistic for target speaker and authentication speech segment using Eq. 3.27.
2. Estimate $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$ for e target speaker and a authentication speech segment, respectively, using Eq. 2.53.
3. Normalize $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$ by their squared-root L^2 forms in order to increase the Gaussianiality of i-vectors as mentioned in [43].

$$\hat{\boldsymbol{\omega}}_n = \frac{\boldsymbol{\omega}_n}{||\boldsymbol{\omega}_n||}, \text{ where } n \in \{i, j\}. \tag{2.57}$$

4. Estimate LLR score either by estimating cosine distance between $\hat{\boldsymbol{\omega}}_i$ and $\hat{\boldsymbol{\omega}}_j$ as described in [26, 28] or by using a PLDA model.

An i-vector contains information not only about the speaker identity but also to a large extent about other factors such as the speaker's emotions, transmission channels, languages, and environmental noises. These other factors in all can be referred to as *inter-session variability* or *channel* factors and should ideally be removed before verification. Three popular channel compensation techniques, namely within class covariance normalization (WCCN) [60], linear discriminate analysis (LDA), and nuisance attribute projection (NAP) [19], were used to remove the effect of channel factors from the i-vectors in [28]. The low dimension of the i-vector inspired researchers to use more advanced methods. Currently, PLDA introduced in [79] has became one of the state-of-the-art methods for removing channel effects from i-vectors in text-independent ASVS.

2.5 Gaussian Mixture Adaptation

When large amounts of speech data is available, MLE using EM estimation algorithm is a good approach to train a speaker specific GMM for each speaker. The necessary amount of speech data depends on how many Gaussian components are there in the GMM. As a *rule-of-thumb*, to estimate the parameters of a distribution which follow a Gaussian distribution, we need at least 30 observations [38]. The more observations we have, the more accurate estimation we can achieve. Therefore, if we want to estimate parameters of a GMM having 512 Gaussian components, we need at least 512*30 or 15360 observations or feature vectors. We can get this number of feature vectors from a 2.56 minutes long speech segment if we extract 1 feature

vector per 10ms speech. However, we cannot guarantee that we will get an equal number of feature vectors (i.e., at least 30) from 2.56 minutes speech for each Gaussian. Most likely many of the feature vectors will be assigned to a small number of Gaussian components. As a result, the parameters of the most of the Gaussian components are not reliably estimated. For this reason, we need many hours of speech data in order to make sure that there will be enough data for estimating the parameters of each Gaussian. However, for real applications, it is unpractical to ask a user to talk for a long time in order to train his/her GMM. To solve this problem in GMM-based speaker verification, a speaker independent GMM is at first trained using several hours of speech by hundreds of speakers. The parameters of this GMM is then adjusted to make a speaker-specific GMM by using a small amount of speech uttered by the speaker. The speaker-independent GMM is called universal background model (UBM) and the parameter adjustment is refereed to as adaptation. Figure 2.6 is a schematic example of making speaker-specific GMM by adapting UBM. The UBM is trained by using speech from many speakers which is then adapted by using shorter speech segments of two speakers to make two speaker-specific GMMs.

All adaptation techniques used in speech processing can be classified into the following two groups:

1. **Non-structural adaptation technique**: It is also known as a static adaptation technique because the number of free parameters is constant regardless of the amount of adaptation data. Relevance maximum-a-posteriori (MAP) [45] and eigenvoice adaptation techniques belong to this category.

2. **Structural adaptation technique**: It first groups the free parameters in a hierarchical manner and depending on the available adaptation data, it controls the number of free parameters. If only a small amount of adaptation data is available, the free parameters of a few major groups are adjusted, and if a reasonable amount of adaptation data is available, then every free parameter is adjusted. Maximum-likelihood-linear-regression (MLLR) [92] and structural MAP (SMAP) [119] adaptation techniques are examples of this type of adaptation.

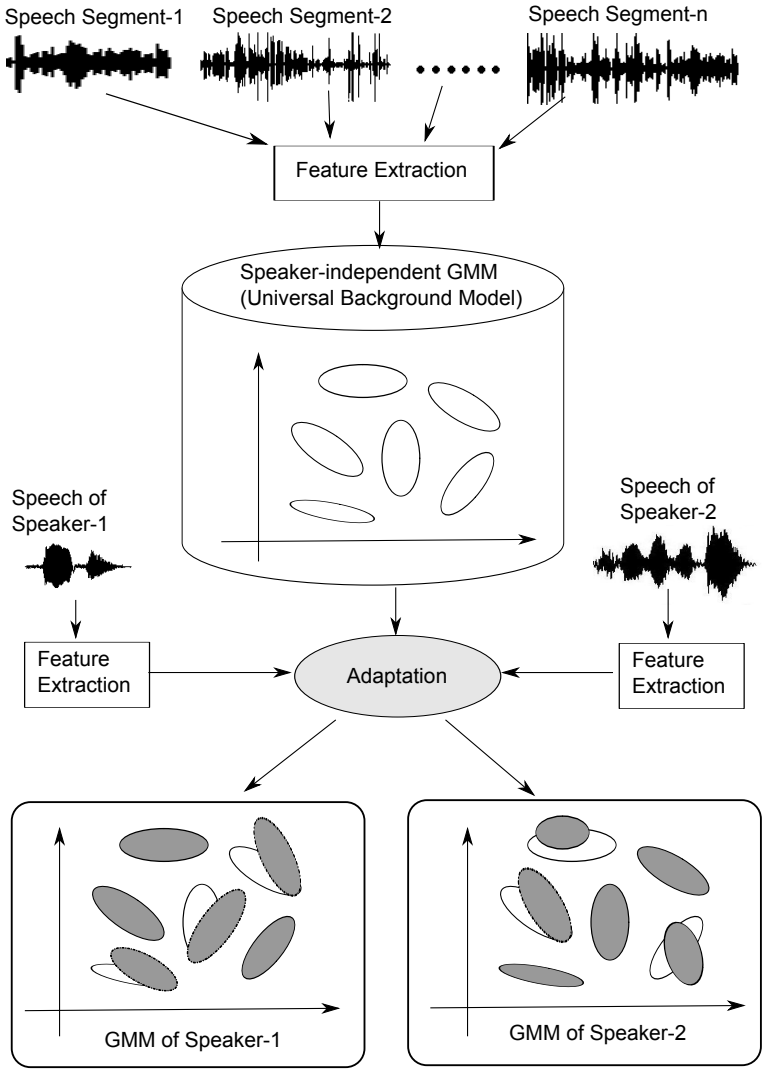

Figure 2.6: An example of UBM adaptation

2.5.1 Maximum-A-Posterirori (MAP) adaptation

This adaptation technique is also known as Bayesian adaptation technique. To adapt a UBM with M Gaussian components, using a set of feature vectors, $X = \{x_1, x_2, ...x_t\}$, the steps of MAP adaptation technique are given below:

1. **Step-1**: Compute the sufficient statistics, that means the number of frames for each component, and the first and second moments as fol-

lows:

$$N_m = \sum_{t=1}^{T} p(m|x_t) \tag{2.58}$$

$$E_m(X) = \frac{1}{N_m} \sum_{t=1}^{T} p(m|x_t) x_t \tag{2.59}$$

$$E_m(X^2) = \frac{1}{N_m} \sum_{t=1}^{T} p(m|x_t) x_t^2 \tag{2.60}$$

where

$$p(m|x_t) = \frac{w_m p_m(x_t)}{\sum_{j=1}^{M} w_j p_j(x_t)} \tag{2.61}$$

2. **Step-2**: Update the old sufficient statistics of UBM for mixture m as follows:

$$\hat{w}_m = [\frac{\alpha_m^a N_m}{T} + (1 - \alpha_m^a) w_m] \beta \tag{2.62}$$

$$\hat{\mu}_m = \alpha_m^b E_m(X) + (1 - \alpha_m^b) \mu_m \tag{2.63}$$

$$\hat{\sigma}_m^2 = \alpha_m^c E_m(X^2) + (1 - \alpha_m^c)(\sigma_m^2 + \mu_m^2) - \hat{\mu}_m^2. \tag{2.64}$$

The scale factor β, is computed over all adapted weights to ensure they sum to unity and the data dependent adaptation coefficient α_m^ρ for $\rho \in \{a, b, c\}$ is defined as:

$$\alpha_m^\rho = \frac{N_m}{N_m + \tau^\rho} \tag{2.65}$$

where τ^ρ is a fixed relevance factor for parameter ρ. Reynolds et al. [114] showed that the mean value adaptation gives the best speaker verification result.

2.5.2 SMAP

Structural maximum-a-posteriori (SMAP) adaptation technique was first proposed by Shinoda et al. [119] for speech recognition. In speaker verification, Liu et al. [95] and Xiang et al. [136] successfully applied it to speech segments of two minutes or shorter.

The SMAP adaptation was proposed to keep the desirable asymptotic properties of relevance MAP while dealing with the problem of data sparseness by using a tree structure. The SMAP-based method have two steps. In the first step, a tree is built by clustering Gaussian components of the UBM.

The root node of the tree represents the whole acoustic space and each of the non-leaf nodes has a Gaussian component that summarizes its child node distributions. Each leaf node corresponds to a Gaussian component in the UBM. In the second step, a speaker-dependent model is obtained by using the distribution of each non-leaf node as the prior for parameters of its child nodes. The adaptation steps for each node p using adaptation data $X = \{x_1, x_2, ..., x_T\}$ are:

1. Transform each sample vector x_t into a vector y_{mt} for each mixture component m as follows:

$$y_{mt}^{(p)} = \mathbf{\Sigma}_m^{-1/2}(x_t - \mu_m^{(p)}), \tag{2.66}$$

   where $t = 1, 2, ..., T$ and $m = 1, 2, ..., M^{(p)}$.
2. Estimate the normalized pdf $\mathcal{N}(Y^{(p)}|\nu, \eta)$ for $Y_m^{(p)} = \{y_{m1}^{(p)}, y_{m2}^{(p)}, ..., y_{mT}^{(p)}\}$, where $\nu^{(p)}$ and $\eta^{(p)}$ represent the shift and rotation needed to compensate for the distortion, i.e., to adapt the model parameters to the data. When there is no mismatch between the training and adaptation data, then $\nu^{(p)} = \vec{0}$ and $\eta^{(p)} = I$. The ML estimation of the mean vector of the normalized pdf is calculated as follows:

$$\tilde{\nu}^{(p)} = \frac{\sum_{t=1}^{T}\sum_{m=1}^{M^{(p)}} \gamma_{mt}^{(p)} y_{mt}^{(p)}}{\sum_{t=1}^{T}\sum_{m=1}^{M} \gamma_{mt}^{(p)}}, \tag{2.67}$$

   where $\gamma_{mt}^{(p)}$ is the occupation probability for Gaussian m at tree node p and time t.
3. Calculate the hierarchical prior

$$\hat{\nu}^{(p)} = \frac{N^{(p)}\tilde{\nu}^{(p)} + \tau\hat{\nu}^{(p-1)}}{N^{(p)} + \tau}, \tag{2.68}$$

   where $N^{(p)} = \sum_{t=1}^{T}\sum_{m=1}^{M^{(p)}} \gamma(p)_{mt}$ is the average number of frames assigned to node pdf p and τ is the MAP relevance factor that weights the priors at the parent node $p-1$.
4. Compute the SMAP estimate of the mean vector

$$\hat{\mu}_m^{(p)} = \mu_m^{(p)} + \mathbf{\Sigma}_m^{1/2}\hat{\nu}^{(p)}, \tag{2.69}$$

   where $\mu_m^{(p)}$ is the unadapted mean vector for Gaussian m of node p.

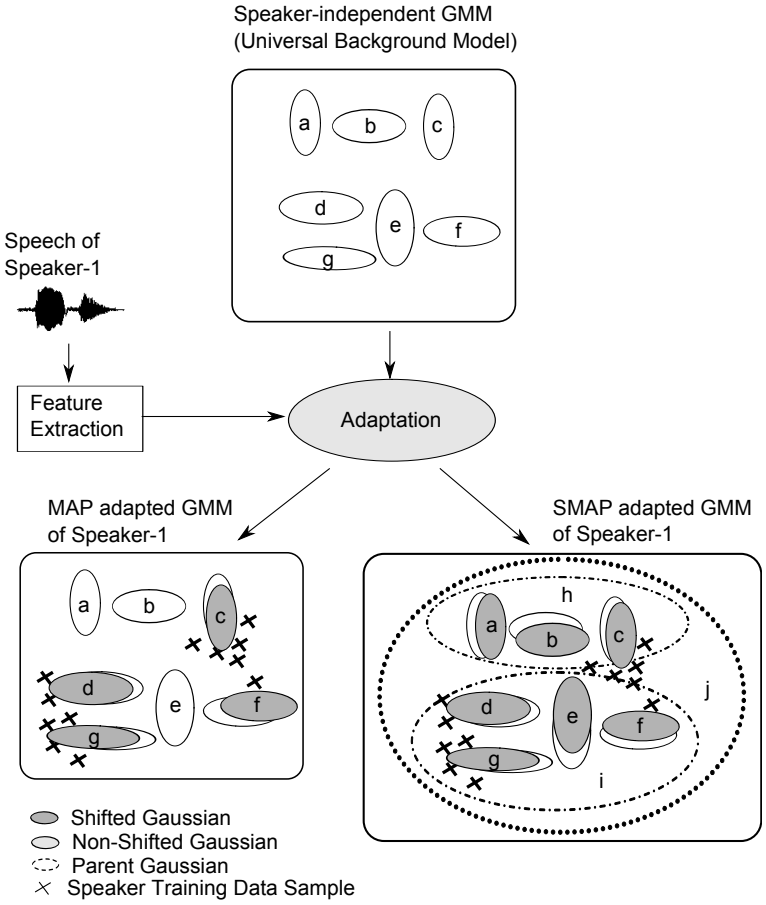

Figure 2.7: Comparison between relevance MAP and SMAP adaptation.

2.5.3 MAP vs SMAP

When no adaptation data is available for a Gaussian component, this Gaussian is not shifted in relevance MAP. In SMAP adaptation, in such case, it takes prior information from its parent Gaussian. Accordingly, every Gaussian component is shifted from its position in UBM. Figure 2.7 shows a schematic example, where in SMAP, $\{a,\ b,\ c\}$ get prior information from h, $\{d, ..., g\}$ from i, and $\{h,\ i\}$ from j. Therefore, every Gaussian component is shifted in SMAP whereas in MAP, Gaussian components $\{a,\ b,\ c\}$ are not shifted.

2.6 Evaluation Metrics

The task of an ASVS is to decide whether speech of the target speaker occurs in an authentication speech, X, and to generate a similarity score. By comparing the similarity score with a threshold, an ASVS takes a decision. The higher the score, the greater confidence that X is spoken by the target speaker. In order to evaluate the performance of an ASVS, a set of trials need to be prepared. A trial consists of a target speaker ID and an authentication speech segment, X. There are two kinds of trials: *target trial* and *non-target trial*. In a *target trial*, X is from a client, whereas in a non-target trial, X is from an impostor. The ultimate goal of an ASVS is to generate low similarity score for a non-target trial and high similarity score for a target trial.

Two commonly used evaluation metrics for an ASVS are the detection cost function (DCF) and the equal error rate (EER). The lower the DCF or EER is, the better the system is. Both evaluation metrics consider two kinds of errors occurred in an ASVS: *false acceptance* (FA) and *false rejection* (FR). First kind of error occurs when the ASVS accepts an impostor's false claim, while the second kind of error occurs when the ASVS rejects a client's true claim. The proportion of these two kinds of errors are known as *false acceptance rate* (FAR) and *false rejection rate* (FRR) which can be written as:

$$\text{FAR} = \frac{\text{FA}}{n_I}, \ \text{FRR} = \frac{\text{FR}}{n_C}, \tag{2.70}$$

where FA, FR, n_I and n_C are the total number of false acceptances, false rejections FR, client accesses, and impostor accesses, respectively.

The DCF has been used as primary evaluation metric in the recent NIST SREs. It assigns one cost for FA (C_{FA}), one cost for FR (C_{FR}), and, a prior probability for a trial being a target trial (P_{tar}). The DCF is the *empirical expected cost* of a trial, estimated on the evaluation data, i.e.,

$$\text{C}_{\text{DCF}} = \text{P}_{\text{tar}}\text{C}_{\text{FR}}\text{P}_{\text{FR|Target}} + (1 - \text{P}_{\text{tar}})\text{C}_{\text{FA}}\text{P}_{\text{FA|Nontarget}}, \tag{2.71}$$

where $\text{P}_{\text{FR|Target}}$ and $\text{P}_{\text{FA|Nontarget}}$ are the empirical error rates for target trials (false rejection rate) and non-target trials (false acceptance rate), respectively, estimated in the evaluation data. Notice that a system that always rejects all trials, will obtain the cost $P_{\text{tar}}\text{C}_{\text{FR}}$ and a system that always accepts all trials will obtain the cost $(1 - \text{P}_{\text{tar}})\text{C}_{\text{FA}}$. Therefore, a useful system

should obtain a DCF that is lower the smallest of these costs. To reflect this, DCF is often normalized as,

$$C_{Norm} = C_{Det} / \min(P_{tar} C_{FR}, (1 - P_{tar}) C_{FA}) \tag{2.72}$$

which then should not be larger than 1 for a useful system. Obviously, DCF depends on the decision threshold, Φ. In order to ignore the problem of threshold tuning, which can be regarded as a separate problem, researchers sometimes use the optimal decision threshold for the evaluation set. The DCF obtained in this way is called *minimum* DCF, C^{min}. As a realistic evaluation metric, *actual detection cost function* (C^{act}) has been embraced by speaker verification community since 2010. C^{act} is the cost estimated by Bayesian decision rule after mapping scores to *log-likelihood ratios* (LLRs).

When some non-target speakers are known, i.e., any of the enrolled speakers and some are completely unknown to the system, DCF can be written as:

$$\begin{aligned} C_{Norm} =& P_{FR|Target} + \beta \times P_{Known} \times P_{FA|KnownNontarget} \\ &+ \beta \times (1 - P_{Known}) \times P_{FA|UnknownNontarget}, \end{aligned} \tag{2.73}$$

where P_{Known} is the *a priori* probability that the non-target speaker is one of the enrolled speakers, and

$$\beta = \frac{C_{FR}}{C_{FA}} \times \frac{P_{tar}}{1 - P_{tar}}, \tag{2.74}$$

When $P_{Known} > 0$, we need to use compound LLRs instead of the original *simple* LLRs. In order for compound LLRs to be effective, it is important that the simple LLRs are well-calibrated. It is not sufficient to calibrate the compound LLRs themselves. Therefore, to simply optimize the decision threshold for the compound LLRs does not give the lowest cost that could have been obtained with perfect calibration. See the BOSARIS website for SRE12 [13] and the materials therein for an explanation about compound vs. simple LLRs.

In order to encourage systems that work well on a large range of operating points (DCFs), the primary evaluation metric, DCF, can be written as the average of two normalized detection costs given by:

$$C_{avg} = \frac{C_{Norm}^{\beta_1} + C_{Norm}^{\beta_2}}{2}, \tag{2.75}$$

where $C_{\text{Norm}}^{\beta_1}$ and $C_{\text{Norm}}^{\beta_2}$ are two different normalized DCFs corresponding to different values of the parameters P_{tar}, C_{FR} and C_{FA}.

Since the DCF is an average of the two error rates (false acceptance and false rejection), it is not particularly intuitive. A more intuitive evaluation metric is the EER. This evaluation metric indicates the number of errors when the decision threshold is set so that FAR and the proportion of false rejections FRR are equal.

2.7 Significance Test

In statistics, a *significance test* is used to decide whether a null hypothesis should be rejected or retained. In speaker verification, it should be used to compare the performances of two systems or the performances of a novel approach to the existing state-of-the art approach on the same problem to determine whether the difference between the measured performance levels of two ASVS is statistically significant. However, in the current literature related to ASVS it has been noticed that most of the time no significant test is done for comparing two systems or two approaches. In this thesis, we follow steps mentioned in [5] for doing significance test of the results of our ASVS. Let two approaces are A and B. Then significance test can be performed by

1. Consider null hypothesis, H_0, and alternative hypothesis, H_a:
   H_0: Approach-B is equally good as Approach-A.
   H_a: Approach-B is better than Approach-A.
2. Estimate z-statistic:

$$z = \frac{|\text{FAR}_{AB} - \text{FAR}_{BA} + \text{FRR}_{AB} - \text{FRR}_{BA}|}{\sqrt{\frac{\text{FAR}_{AB}+\text{FAR}_{BA}}{4 \cdot n_I} + \frac{\text{FRR}_{AB}+\text{FRR}_{BA}}{4 \cdot n_C}}} \tag{2.76}$$

   where n_I and n_C are the total number of impostor accesses and client accesses. FAR_{AB} is the proportion of impostor accesses correctly rejected by Approach-A based ASVS, but mistakenly accepted by Approach-B based ASVS. On the other hand, FAR_{BA} is the proportion of impostor accesses correctly rejected by Approach-B based ASVS, but mistakenly accepted by Approach-A based ASVS. Same explanation can be applied for FRR_{AB} and FRR_{BA} .

3. Set confidence level, α and determine the value of $Z_{\alpha/2}$ from the z-Table.
4. If $z > Z_{\alpha/2}$, reject the null hypothesis, H_0 and take a decision that Approach-B is statistically significantly better than Approach-A

In this approach, data dependency issue is ignored. Data dependency in an ASVS arises largely from multiple uses of the same speakers in order to provide more target and non-target trials due to limited resources. Generally, the number of target speakers and the number of authentication speeches are very small. By using the same target speakers and authentication speeches, a set of trials are prepared for evaluating any system which causes data dependency. This data dependency issue is complicated. Rohdin et al. [115], dealt this issue for discriminant training of a PLDA model. However, for significance test, no suitable approach has been found in the current literature. Therefore, data dependency has been overlooked in this thesis.

Chapter 3

Related Work

In this chapter, we give an overview of previous works on intersession variability, short duration of utterance and background models. The organization of this chapter is as follows: Section 3.1 and Section 3.2 point out some approaches proposed to deal with inter-session variability and short utterances, respectively. Section 3.3 discusses previous works on robust background models.

3.1 Inter-session Variability Compensation

Except the context of speech, factors which change the characteristics of a speaker's speech signal from time to time are known as *inter-session variability* or *channel variability* factors. For example, the acoustic environment in which a system is operating, the speaker's emotional state and physical conditions, the input equipments, the transmission channels and so on are considered as inter-session variability factors because they add information to the speech signal which is uncorrelated to the identity of the speaker. Because of these factors, an authentication utterance may sound different from the training utterance(s) even though they are spoken by the same speaker, as a result the verification performance degrades a lot. The process of separating these factors from the speaker identity or mitigating the effect of these factors is referred to as *channel compensation* or *inter-session variability compensation* which has become one of the core tasks of ASVS. Various approaches have been proposed at almost every stage in an ASVS which can be divided mainly into three categories: feature space based compensation,

model-based compensation, and score normalization. Both feature space based compensation and score normalization can be used with a wide variety of classifiers, whereas model-based approaches are less general since they are specific to a particular classifier. A few approaches for each of these three categories are listed below:

- **1974**: Cepstral mean subtraction (CMS) was proposed by Atal [3].
- **1996**: Handset-dependent score normalization (H-Norm) was proposed by Reynolds [112].
- **1998**: Modulation spectral analysis based approach was applied by Vuuren et al. [127].
- **2000**: Speaker model synthesis (SMS) and test normalization (T-Norm) were proposed by Teunen et al. [126] and Auckenthaleret et al. [4], respectively.
- **2001**: Feature warping was proposed by Pelecanos et al. [106].
- **2003**: Feature mapping was proposed by Reynolds [113].
- **2004**: Joint factor analysis (JFA) was proposed by Kenny et al. [83] for GMM-UBM based ASVS.
- **2006**: Nuisance attribute projection (NAP) and within-class covariance normalization (WCCN) were proposed by Campbell et al. [19] and Hatch [60], respectively, for the GMM-SVM based ASVS.
- **2009**: WCCN, LDA and NAP were used by Dehak et al. [26] for i-vector based ASVS.
- **2010**: Heavy-tailed PLDA was proposed by Kenny in [79] for i-vector based ASVS.
- **2011**: Length normalized Gaussian PLDA (G-PLDA) was introduced by Garcia-Romero et al. [43] for i-vector based ASVS.
- **2014**: Weighted maximum margin criterion (WMMC) and source-normalized WMMC (SN-WMMC) were introduced by Kanagasundaram et al. [75].

Among the above mentioned approaches, CMS, RASTA filtering, feature warping and feature mapping are feature space based compensation techniques, H-norm and T-norm are score normalization techniques and the others are model based compensation techniques. These approaches are described briefly in Sections 3.1.1- 3.1.3. A comparative study about their performances is done in Section 3.1.4.

3.1.1 Feature Space-based Compensation

CMS is a very simple technique, which can achieve a considerable amount of robustness against disturbing channel and speaker effects. In this approach, the mean of the cepstral vectors of an utterance is subtracted from each frame of that utterance:

$$y[t] = x[t] - \frac{1}{T}\sum_{i=1}^{T} x[i] \tag{3.1}$$

where $x[t]$ and $y[t]$ are the time-varying cepstral vectors of the utterance before and after CMS, respectively, the index 't' refers to the analysis frames and T is the total number frames in an utterance. This approach not only reduces linear channel effects, but also reduces the average vocal tract configuration information of the speaker. Therefore, under different channel environments, it is beneficial to apply CMS, but applying it on clean speech with matched transducer and channel conditions degrades performance [110].

Modulation spectrum based approaches were proposed to suppress the spectral components that change more slowly or quickly than the typical range of change of speech. RelAtive SpecTrA (RASTA) filtering [65] is such a popular approach which was proposed for robust speech recognition and accepted by research community worked on speaker verification due to its simplicity and good performance for removing convolutional noise (e.g., noise caused by recording devices such as microphone) as well as additive noise (e.g., car passing sound, air-conditioned sound etc). In RASTA filtering, at first the critical band power spectrum is computed from each analysis frame. Then a compressing static non-linear transformation is applied on the spectral amplitude. After that, the time trajectory of each transformed spectral component is filtered out and then the filtered speech representation is transformed through an expanding non-linear transformation. Then the steps of conventional PLP (See Section 2.3.4) can be applied to simulate power law of hearing and compute an all-pole model.

The RASTA filter has a bandpass about 1 to 13 Hz. This is suitable for speech recognition, but its lower cut-off frequency removes significant portions of speaker specific information, and higher cut-off frequency includes unnecessary information. By computing relative importance of different spectral components of the magnitude spectrum, Vuuren et al. [127] showed that spectral components from 0.1 Hz to 10 Hz contain the most

useful speaker information. They designed a filter appropriate for speaker verification based on their findings. In their approach, static features are derived by applying CMS and 101-tap lowpass FIR filter with cut-off at 10 Hz to the time sequences of cepstrums. Dynamic features are derived by applying a delta filter to the static features. Both the static and dynamic features are downsampled to 25 Hz.

In feature warping [106], at first a set of cepstral co-efficients are extracted from the speech segment. Then the short term distribution of the individual cepstral co-efficients is conditioned to a standardized distribution. For this, a sliding window (typically, a three second window) is shifted by a single frame each time by keeping the cepstral co-efficient whose warped value will be estimated in the center of the sliding window. The co-efficient values inside the sliding window are sorted in descending order and their ranking within the sorted list is calculated. The most positive co-efficient obtains a ranking of 1 while the most negative a ranking of the size of window, N. This ranking is used as an index in a lookup table to determine the warped feature value. The lookup table used to perform the mapping is calculated prior to the parameterization process using

$$\int_{z=-\infty}^{m} h(z)dz = \frac{N + \frac{1}{2} - R}{N} \tag{3.2}$$

where z is a single warped feature stream variable, $h(z)$ is the target density function of z, N is the size of the sliding window, R is the rank of the middle speech feature of the current window and m is the warped feature value. If the target density function is a normal distribution, then the warped feature is estimated by setting the rank initially to $R = N$, solving for m by numerical integration, and repeating for each decremented value of R for the following equation:

$$\int_{z=-\infty}^{m} \frac{1}{\sqrt{2\pi}} \exp\Big(-\frac{z^2}{2}\Big)dz = \frac{N + \frac{1}{2} - R}{N} \tag{3.3}$$

3.1.2 Score Normalization

In *handset-dependent score normalization*, the basic approach is to estimate handset-dependent biases, μ, and scales, σ, in the LLR scores from a set of

impostors and then remove these from scores of target speakers during verification. For that, at first a set of handsets, H, are decided whose scores will be normalized. Then a set of impostors, $\mathcal{I}$, are selected from a background dataset, $\mathcal{B}$, which have recordings for each handset, $h \in H$. After that a set of impostor models, $\mathcal{I}_h$, are trained. Then a set of speech segments, X_h, are selected as test segments for each $h \in H$. After that LLR score is estimated for each $\mathcal{I}_h$-UBM model pair using X_h. Then μ_h and σ_h are estimated which is used in score normalization. To avoid bimodal distributions, gender information is matched. The H-normalized score, $\hat{\mathcal{S}}_x^t$, of a LLR score, $\mathcal{S}_x^t$, for a given target speaker t and an authentication utterance x, can be defined as:

$$\hat{\mathcal{S}}_x^t = \frac{\mathcal{S}_x^t - \mu_{h(x)}}{\sigma_{h(x)}}, \tag{3.4}$$

where $h(x)$ is the handset label for the authentication utterance x. In this approach, a handset label detector is necessary when meta data about handset label is not available. This approach does not work on a unknown handset. Therefore it did not get popularity.

Test normalization (T-norm) is applied to transform the log likelihood ratio (LLR) score of an ASVS in order to reduce inter-session variability associated with the LLR score [4]. Unlike H-Norm, it does need to detect handset label and to have a set of impostors for each handset. The T-normalized score, $\hat{\mathcal{S}}_x^t$, of a LLR score, $\mathcal{S}_x^t$, for a given target speaker t and an authentication utterance x, can be defined as:

$$\hat{\mathcal{S}}_x^t = \frac{\mathcal{S}_x^t - \mu}{\sigma}, \tag{3.5}$$

where parameters μ and σ are estimated as the sample mean and standard deviation, respectively, of a set of LLR scores generated by impostor models for the authentication utterance t. Impostors used in T-normalization are known as T-norm speakers.

3.1.3 Model Based Compensation

Speaker Model Synthesis (SMS)

In this approach, speaker-independent channel transformations are estimated offline, which are then used to synthesize speaker models for channels for which no speaker data is available. For an authentication utterance,

first caller's channel is identified by a channel detector. If the detected channel is different from the target speaker's channel, then a synthesized model is built for the target speaker by applying the appropriate transformation. Likelihood scoring and normalization is performed using the synthesized model and channel-specific UBM.

Given a gender-, channel- and speaker-independent root UBM, $\lambda_r(w_r, \mu_r, \sigma_r)$, and a set of feature vectors $\{x_i\}_{i=1}^{N}$ from a channel, $h \in 1, 2, ..., H$, a channel- and gender-dependent UBM can be obtained by adapting the c-th Gaussian component of the root UBM, λ_r, in the following way:

$$w_{h,c} = \alpha \frac{n_c}{N} + (1 - \alpha) w_{r,c} \tag{3.6}$$

$$\mu_{h,c} = \alpha \left(\frac{1}{n_c} \sum_{i=1}^{N} p(c|x_i, \lambda_r) x_i \right) + (1 - \alpha) \mu_{r,c} \tag{3.7}$$

$$\sigma_{h,c}^2 = \alpha \left(\frac{1}{n_c} \sum_{i=1}^{N} p(c|x_i, \lambda_r) x_i^2 \right) + (1 - \alpha)(\sigma_{r,c}^2 + \mu_{r,c}^2) - \mu_{h,c}^2 \tag{3.8}$$

where α is the smoothing factor and n_c is defined as:

$$n_c = \sum_{i=1}^{N} p(c|x_i, \lambda_r). \tag{3.9}$$

After training channel- and gender-dependent UBM, λ_h, channel is detected for the enrollment data of a target speaker. After that the corresponding λ_h is adapted by the enrollment data to make channel- and speaker-dependent model, $\lambda_s(w_s, \mu_s, \sigma_s)$. In the authentication phase, if the channel of authentication segment is different from the channel of target speaker, then a transformation is done from channel a to channel b denoted by T_{ab} in the following way:

$$T_{ab}(w_{s,c}) = w_{s,c} \left(\frac{w_{b,c}}{w_{a,c}} \right) \tag{3.10}$$

$$T_{ab}(\mu_{s,c}) = \mu_{s,c} + (\mu_{b,c} - \mu_{a,c}) \tag{3.11}$$

$$T_{ab}(\sigma_{s,c}^2) = \sigma_{s,c}^2 \left(\frac{\sigma_{b,c}^2}{\sigma_{a,c}^2} \right) \tag{3.12}$$

Within-Class Covariance Normalization (WCCN)

In WCCN [60], first a set of upper bounds on the rates of false positives and false negative at a given score threshold are constructed. Under various conditions, minimizing these bounds leads to the closed-form solution, $\mathbf{W}^{-1}$, where $\mathbf{W}$ is the expected within-class covariance matrix of the data, given by:

$$\mathbf{W} = \sum_{c=1}^{C} p(c)\mathbf{\Sigma}_c \tag{3.13}$$

$$\mathbf{\Sigma}_c = \mathrm{E}(\boldsymbol{x}_c - \bar{\boldsymbol{x}}_c)(\boldsymbol{x}_c - \bar{\boldsymbol{x}}_c)', \forall_c \tag{3.14}$$

where C is the total number of classes, $p(c)$ is the prior probability of class c and $\boldsymbol{x}_c$ is a random draw from class c in input space. After estimating $\mathbf{W}$, a generalized linear kernel can be obtained as:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i' \mathbf{W} \boldsymbol{x}_j \tag{3.15}$$

If $\mathbf{W}$ is full-rank, then it can be written as:

$$\mathbf{W}^{-1} = \mathbf{U}\mathbf{U}' \tag{3.16}$$

For a small set of feature vectors, generally $\mathbf{W}$ is estimated empirically. In practice, empirical estimates of $\mathbf{W}$ are typically quite noisy. In order to denoise empirically estimated $\hat{\mathbf{W}}$, generally a smoothing technique is applied. In [59] the following smoothing model is used:

$$\hat{\mathbf{W}}_s = (1-\alpha) \cdot \hat{\mathbf{W}} + \alpha \cdot \mathbf{I},\ \alpha \in [0, 1] \tag{3.17}$$

For large feature sets, inverting or simply estimating $\hat{\mathbf{W}}$ is impractical for computational reasons. To deal with this problem, in [59] the PCA decomposition described in [73] was proposed to use, where the feature space is divided into two sets: a PCA-set and a PCA-complement set.

Nuisance Attribute Projection (NAP)

NAP [19] was proposed mainly for SVM based system, but it is enable to work with any other classifiers. In this approach, a projection matrix, $\mathbf{P}$ is learned to remove the components of a vector (e.g., supervector) in the direction of a subspace that contains mostly channel information rather than

information about speakers identity. The filtered out components are called *nuisance attributes*. Projection matrix, **P**, is defined as:

$$\mathbf{P} = \mathbf{I} - \mathbf{U}\mathbf{U}' \tag{3.18}$$

where **U** is a matrix with orthonormal columns. Thousands of speakers recorded in multiple conditions are used to train **P**. Let S be the number of speakers and H_s be the number of utterances from s-th speaker. By extracting $MD \times 1$ dimensional mean vector $\boldsymbol{\mu}$ from each session $h = 1, 2, ..., H_s$ of each speaker $s = 1, 2, ..., S$, the projection matrix **P** is estimated by the following objective function

$$\text{Minimize} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} w_{ij} ||\mathbf{P}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)||^2 \tag{3.19}$$

$$\text{subject to } \mathbf{U}'\mathbf{U} = \mathbf{I} \tag{3.20}$$

where $N = \sum_{s=1}^{S} H_s$ and w_{ij} is a weight which can be defined as:

$$w_{ij} = \begin{cases} 1 & \text{if } \boldsymbol{\mu}_i \text{ and } \boldsymbol{\mu}_j \text{ are from the same speaker} \\ 0 & \text{otherwise} \end{cases}$$

The objective function 3.19 is minimized by setting the columns of **U** be the d most principal eigenvectors of the eigenvalue problem

$$\mathbf{A}\mathbf{Z}\mathbf{A}'\mathbf{U} = \mathbf{U}\Lambda \tag{3.21}$$

where $\mathbf{A} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2....., \boldsymbol{\mu}_N]$ and **Z** is defined as:

$$\mathbf{Z} = \text{diag}(\mathbf{W} \cdot \mathbf{1}) - \mathbf{W} \tag{3.22}$$

where $\mathbf{W} = [w_{ij}]$ is a $N \times N$ symmetric matrix and **1** is a vector of ones.

Joint Factor Analysis (JFA)

JFA [83, 78, 80] was proposed for speaker recognition using the GMM-supervector. In this approach, it is assumed that a speaker- and channel-dependent GMM-supervector, $\boldsymbol{\mu}_s^c$, can be decomposed into a speaker-dependent part, $\boldsymbol{s}$, and a channel-dependent part, $\boldsymbol{c}$, in the following way:

$$\boldsymbol{\mu}_s^c = \boldsymbol{s} + \boldsymbol{c}, \tag{3.23}$$

where

$$\boldsymbol{s} = \boldsymbol{\mu} + \mathbf{V}\mathbf{y} + \mathbf{D}\mathbf{z}, \tag{3.24}$$

$$\boldsymbol{c} = \mathbf{U}\mathbf{x}, \tag{3.25}$$

where

- $\boldsymbol{\mu} \in \mathbb{R}^{MD\times 1}$ is a speaker- and channel-independent GMM-supervector representative of D-dimensional M number of mean vectors of an UBM.
- $\mathbf{V} \in \mathbb{R}^{MD\times R_s}$ is the eigenvoice matrix with rank R_s, where $R_s \ll MD$.
- $\mathbf{U} \in \mathbb{R}^{MD\times R_c}$ is the eigenchannel matrix with rank R_c, where $R_c \ll MD$.
- $\mathbf{D} \in \mathbb{R}^{MD\times MD}$ is the diagonal residual matrix with full-rank.
- $\mathbf{y} \in \mathbb{R}^{R_s\times 1}$, $\mathbf{x} \in \mathbb{R}^{R_c\times 1}$ and $\mathbf{z} \in \mathbb{R}^{MD\times 1}$ are hidden variables distributed according to the standard normal distribution, $\mathcal{N}(\mathbf{y}|\mathbf{0},\mathbf{I})$, $\mathcal{N}(\mathbf{x}|\mathbf{0},\mathbf{I})$ and $\mathcal{N}(\mathbf{z}|\mathbf{0},\mathbf{I})$, respectively. They are called *speaker factors*, *channel factors* and *speaker-dependent residual factors* or *common factors*, respectively.
- $\boldsymbol{s}$ is normally distributed with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{V}\mathbf{V}' + \mathbf{D}^2$, and $\boldsymbol{c}$ is normally distributed with zero mean and covariance matrix $\mathbf{U}\mathbf{U}'$.
- The range of $\mathbf{V}\mathbf{V}'$ is known as *speaker space* and the range of $\mathbf{U}\mathbf{U}'$ is known as *channel space*.

For a UBM, $\lambda = \{w_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M}$, and a background dataset, $\mathcal{B}$ with S number of speakers having H_s number of enrollment data per speaker, $s \in \{1, 2, ..., S\}$, hyperparameters $(\mathbf{V}, \mathbf{U}, \mathbf{D})$ are estimated in a iterative process. At first eigenvoice matrix $\mathbf{V}$ is estimated assuming that $\mathbf{U}$ and $\mathbf{D}$ are zero, then using the estimated $\mathbf{V}$ eigenchannel matrix $\mathbf{U}$ is estimated assuming that $\mathbf{D}$ is zero, and using estimated $\mathbf{V}$ and $\mathbf{U}$, residual matrix $\mathbf{D}$ is estimated. The steps of estimating $\mathbf{V}$ are given below:

1. Extract D dimensional feature vectors from all utterances of background set, $\mathcal{B}$.
2. For each speaker s with T feature vectors $\{\boldsymbol{o}_1, \boldsymbol{o}_2, ..., \boldsymbol{o}_T\}$ extracted from H_s number of utterances from s, estimate the 0th, 1st and 2nd order Baum-Welch statistics, $N_m(s)$, $\tilde{\mathbf{F}}_m(s)$ and $\tilde{\mathbf{S}}_m(s)$, respectively,

using the m-th Gaussian component of UBM as follows:

$$\begin{aligned} N_m(s) &= \sum_{t \in s} \gamma_{mt} \\ &= \sum_{t \in s} \frac{w_m \mathcal{N}(\boldsymbol{o}_t \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}{\sum_{i=1}^{M} w_i \mathcal{N}(\boldsymbol{o}_t \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \end{aligned} \tag{3.26}$$

$$\tilde{\mathbf{F}}_m(s) = \sum_{t \in s} \gamma_{mt} \boldsymbol{o}_t \tag{3.27}$$

$$\tilde{\mathbf{S}}_m(s) = \text{diag}\Big(\sum_{t \in s} \gamma_{mt} \boldsymbol{o}_t \boldsymbol{o}_t' \Big) \tag{3.28}$$

The 2nd order statistic is unnecessary if $\boldsymbol{\Sigma}$ does not need to be updated.

3. Center 1st and 2nd-order Baum-Welch statistics:

$$\mathbf{F}_m(s) = \tilde{\mathbf{F}}_m(s) - N_m(s) \boldsymbol{\mu}_m, \tag{3.29}$$

$$\begin{aligned} \mathbf{S}_m(s) =& \tilde{\mathbf{S}}_m(s) \\ &- \text{diag}\Big(\tilde{\mathbf{F}}_m(s) \boldsymbol{\mu}_m' + \boldsymbol{\mu}_m \tilde{\mathbf{F}}_m(s)' - N_m(s) \boldsymbol{\mu}_m \boldsymbol{\mu}_m' \Big), \end{aligned} \tag{3.30}$$

4. Decide a desired rank R_s for the eigenvoice matrix, $\mathbf{V}$, and initialize it randomly.
5. Estimate precision matrix, $\mathbf{L}(s)$as follows:

$$\mathbf{L}(s) = \mathbf{I} + \sum_{m=1}^{M} N_m(s) \mathbf{V}_m' \boldsymbol{\Sigma}_m^{-1} \mathbf{V}_m \tag{3.31}$$

where $\mathbf{V}_m \in \mathbb{R}^{D \times R_s}$ is the m-th sub-matrix of $\mathbf{V}$.

6. Estimate speaker factors of speaker s as follows:

$$\mathbf{y}(s) = \mathbf{L}^{-1}(s) \sum_{m=1}^{M} \mathbf{V}_m' \boldsymbol{\Sigma}_m^{-1} \mathbf{F}_m(s) \tag{3.32}$$

7. Estimate accumulators, $\mathbf{C}$ and $\mathbf{A}$ as follows:

$$\mathbf{C} = \sum_{s \in \mathcal{B}} \mathbf{F}(s) \mathbf{y}(s)^{'} \tag{3.33}$$

$$\mathbf{A}_m = \sum_{s \in \mathcal{B}} N_m(s) (\mathbf{L}^{-1}(s) + \mathbf{y}(s) \mathbf{y}(s)^{'}) \tag{3.34}$$

where

$$\mathbf{F}(s) = \begin{pmatrix} \mathbf{F}_1(s) \\ \mathbf{F}_2(s) \\ \vdots \\ \mathbf{F}_m(s) \end{pmatrix} \tag{3.35}$$

8. Update $\mathbf{V}_m$ as follows:

$$\mathbf{V}_m = \mathbf{C}\mathbf{A}_m^{-1} \tag{3.36}$$

9. Update covariance matrix:

$$\mathbf{\Sigma} = \mathbf{N}^{-1}\Big(\sum_{s\in\mathcal{B}} \mathbf{S}(s) - \text{diag}(\mathbf{C}\mathbf{V}')\Big) \tag{3.37}$$

where

$$\mathbf{N} = \sum_{s\in\mathcal{B}} \mathbf{N}(s) \tag{3.38}$$

$$\mathbf{N}(s) = \begin{pmatrix} N_1(s)\cdot\mathbf{I} & 0 & \cdots & 0 \\ 0 & N_2(s)\cdot\mathbf{I} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & N_m(s)\cdot\mathbf{I} \end{pmatrix} \tag{3.39}$$

$$\mathbf{S}(s) = \begin{pmatrix} \mathbf{S}_1(s) & 0 & \cdots & 0 \\ 0 & \mathbf{S}_2(s) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{S}_m(s) \end{pmatrix} \tag{3.40}$$

This step is optional. Most of the time UBM's covariance matrix is used.

10. Iterate Step 5-Step 9 until $\mathbf{V}$ converges.

Steps of estimating $\mathbf{V}$ are almost the same to the steps of estimating i-vector extractor, $\mathbf{T}$ (See Section 2.4.4). The only difference is that for $\mathbf{T}$ all utterances from a speaker are considered as being uttered by different persons, whereas for $\mathbf{V}$ all the utterances of a given speaker are considered to belong to the same person [28].

After computing $\mathbf{V}$, Eq. 3.32 is used to estimate speaker factors, $\mathbf{y}(s)$, for each speaker. Then the 1st-order statistic for each utterance, u of $s \in \mathcal{B}$ is modified as follows:

$$\mathbf{F}_m(u,s) = \tilde{\mathbf{F}}_m(u,s) - N_m(u,s)\Big(\boldsymbol{\mu} + \mathbf{V}\mathbf{y}(s)\Big) \tag{3.41}$$

where

$$\tilde{\mathbf{F}}_m(u,s) = \sum_{t \in u,s} \gamma_{mt} \boldsymbol{o}_t \tag{3.42}$$

$$N_m(u,s) = \sum_{t \in u,s} \gamma_{mt} \tag{3.43}$$

Using $N_m(u,s)$ and $\mathbf{F}_m(u,s)$ instead of $N_m(s)$ and $\mathbf{F}_m(s)$, $\mathbf{U}$ is estimated following the same steps followed for estimating $\mathbf{V}$. To estimate $\mathbf{D}$, the following first order statistic is used in the steps used for estimating $\mathbf{V}$:

$$\mathbf{F}_m(s) = \tilde{\mathbf{F}}_m(s) - N_m(s)\Big(\boldsymbol{\mu} + \mathbf{V}\mathbf{y}(s)\Big) - \sum_{u \in s} N_m(u,s)\mathbf{U}\mathbf{x}(u,s) \tag{3.44}$$

After estimating $\mathbf{V}$, $\mathbf{U}$ and $\mathbf{D}$ using $\mathcal{B}$, $\mathbf{y}$, $\mathbf{x}$ and $\mathbf{z}$ are estimated for the enrollment data of the target speaker and for the authentication data in order to generate score for a trial. There are many scoring strategies. In [47], it has been shown that performance does not vary too much among different scoring techniques, however, the speed of evaluation varies a lot from one technique to another. The fastest scoring technique is the linear scoring which can be written as for a target speaker (tar) and an authentication speech ($test$):

$$\mathcal{S} = \Big(\mathbf{V}\mathbf{y}(tar) + \mathbf{D}\mathbf{z}(tar)\Big)' \boldsymbol{\Sigma}^{-1} \Big(\mathbf{F}(tst) - \mathbf{N}(tst)\boldsymbol{\mu} - \mathbf{N}(tst)\mathbf{U}\mathbf{x}(tst)\Big) \tag{3.45}$$

Probabilistic Linear Discriminant Analysis (PLDA)

PLDA was originally proposed for object recognition in image processing independently by Ioffe [70] and Simon et al. [107]. In [107] it is assumed that the feature vector, $\mathbf{g}$, is generated as:

$$\mathbf{g} = \mathbf{m} + \mathbf{V}\mathbf{y} + \mathbf{U}\mathbf{x} + \boldsymbol{\epsilon}, \tag{3.46}$$

where $\mathbf{m}$ is the mean of $\mathbf{g}$, and $\mathbf{y}$ and $\mathbf{x}$ are random vectors dependent on the class and channel factors, respectively. The vector $\boldsymbol{\epsilon}$ also depends on the channel factors and follows $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is a diagonal covariance matrix. The vectors $\mathbf{y}$ and $\mathbf{x}$ follow the standard normal distribution. The matrix $\mathbf{V}$ is a basis for the *between-class subspace* and the matrix $\mathbf{U}$ is a basis for the *within-class subspace*. This PLDA model is very similar to the JFA model. The difference is that in the PLDA model, $\mathbf{g}$ is observed whereas in JFA, $\boldsymbol{\mu}$ is *indirectly observed*, i.e., we observe features drawn from the GMM but we do not know the parameters of $\boldsymbol{\mu}$.

Kenny introduced PLDA as in Eq. (3.46) for speaker verification with i-vectors as features in [79]. The author suggested to skip $\mathbf{Ux}$ but instead use full covariance $\mathbf{\Sigma}$ when large amounts of data are available, i.e.,

$$\boldsymbol{\omega} = \mathbf{m} + \mathbf{V}\mathbf{y} + \boldsymbol{\epsilon}. \tag{3.47}$$

Using a full covariance matrix, $\mathbf{\Sigma}$, is possible since the dimension of the i-vector is low. The PLDA model in Eq. (3.47) is similar to the *two-covariance model* proposed in [14] and to the PLDA model proposed in [70]. The rank of $\mathbf{V}$ is lower than the dimension of the feature vector in [70, 79]. On the other hand, in [14], the rank of $\mathbf{V}$ is equal to the dimension of the feature vector, which means that the between-class covariance $\mathbf{V}\mathbf{V}^T$ has a full rank.

Given the two i-vectors, $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$ involved in a trial, the verification score, s_{ij}, is computed as:

$$s_{ij} = \log \frac{p(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j|\mathcal{H}_s)}{p(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j|\mathcal{H}_d)}, \tag{3.48}$$

where $\mathcal{H}_s$ and $\mathcal{H}_d$ are the following two hypotheses
$\mathcal{H}_s$: $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$ belong to the same speaker
$\mathcal{H}_d$: $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$ belong to two different speakers

When $\mathbf{m} = \mathbf{0}$, the closed-form solution of Eq. (3.48) is

$$s_{ij} = 2\boldsymbol{\omega}_i^T \mathbf{P} \boldsymbol{\omega}_j + \boldsymbol{\omega}_i^T \mathbf{Q} \boldsymbol{\omega}_i + \boldsymbol{\omega}_j^T \mathbf{Q} \boldsymbol{\omega}_j + c, \tag{3.49}$$

where c is a constant, and

$$\mathbf{P} = \mathbf{\Sigma}_a^{-1}\mathbf{\Sigma}_b - (\mathbf{\Sigma}_a - \mathbf{\Sigma}_b\mathbf{\Sigma}_a^{-1}\mathbf{\Sigma}_b)^{-1}, \tag{3.50}$$

$$\mathbf{Q} = \mathbf{\Sigma}_a^{-1} - (\mathbf{\Sigma}_a - \mathbf{\Sigma}_b\mathbf{\Sigma}_a^{-1}\mathbf{\Sigma}_b)^{-1}, \tag{3.51}$$

where $\mathbf{\Sigma}_a = \mathbf{V}\mathbf{V}^T + \mathbf{\Sigma}$, and $\mathbf{\Sigma}_b = \mathbf{V}\mathbf{V}^T$ [see [43]].

Typically, $\mathbf{V}$ and $\mathbf{\Sigma}$ are estimated by maximizing the likelihood (ML) of the training data

$$[\hat{\mathbf{V}}, \hat{\mathbf{\Sigma}}] = \arg\max_{\mathbf{V},\mathbf{\Sigma}} \{p(\mathcal{B}|\mathbf{V}, \mathbf{\Sigma})\}, \tag{3.52}$$

where $\hat{\mathbf{V}}$ and $\hat{\mathbf{\Sigma}}$ are the ML estimates of $\mathbf{V}$ and $\mathbf{\Sigma}$, respectively, and $\mathcal{B}$ is the set of background i-vectors.

This is usually achieved by means of an EM algorithm [107]. It has been shown in [43] that it is better to apply whitening followed by length normalization to the i-vectors before estimating the parameters of the PLDA model in order to make the i-vectors more closely follow a Gaussian distribution.

3.1.4 Comparisons Among Different Approaches

In [15], a comparative study was done on the effect of CMS, RASTA filtering, heteroscedastic linear discriminant analysis (HLDA) [90], feature warping, feature mapping, eigenchannel adaptation and score normalization. It was shown that doing RASTA filtering after applying CMS can reduce EER almost half comparing to the CMS based system. Feature warping, HLDA and feature mapping on the top of the RASTA filtering was also advantageous. The most beneficial was to do eigenchannel adaptation. It gave an impressive improvement in system's performance especially when it was applied without feature mapping. T-norm was not effective to improve the overall system's performance. Comparing to RASTA filtering feature warping was more beneficial. The baseline system's EER and C^{min} for the NIST SRE 2006 core task (English) were 23.8% and 0.088, respectively. Using all channel compensation techniques along with double delta and triple delta, 4.0% EER and 0.018 C^{min} were obtained. Excluding RASTA, feature mapping and T-norm, EER and C^{min} became 3.6% and 0.018, respectively. In [27], comparison between JFA and SVM with NAP is presented. It was shown that JFA is better successful than NAP based SVM. For the NIST SRE 2006 core task (English), 4.4% EER and 0.024 C^{min} was obtained in SVM based system using non-linear kernel and applying NAP, whereas 3.5% EER and 0.021 C^{min} was obtained in JFA based system.

3.2 Short Duration of Utterance

There is no doubt that the performance of a text-independent ASVS has been improved a lot during the last one and half decades mainly because of the improvement of classifiers and channel compensation techniques. However, all these techniques demand substantial amount of speech both in the training and authentication phases which contradict with the users' demands and reality. Many cooperative users (i.e. users who willingly use an ASVS for their own benefit) do not want to talk happily for a long time to be authenticated by an ASVS even for not frequently used applications (e.g., accessing online banking service). For frequently accessed applications (e.g., login to smart phone, e-mail account) shorter speech gets their higher priority because of convenience. For some applications where users

are unaware about the authentication process like in forensics, it is not possible to ensure that substantial amount of quality speech can be collected for the verification purpose. For some devices like mobile phones, it is hard to use long utterances because of the shortage of memory. Therefore, reducing the required amount of speech data has recently become a focusing point in the state-of-the-art speaker verification design, including GMM-UBM, JFA, SVM and i-vector based systems [100, 35, 130, 81, 103, 76, 77, 84, 74, 75].

In [103], a comparative figure was drawn about the performances of the GMM-SVM based systems using limited speech. Comparing to the GMM-UBM based system, the performance of GMM-SVM based system degraded more rapidly when speech duration was reduced. The duration of impostor utterances had a considerable effect on the system's performance. Matching the impostor utterances to the length of the short authentication utterance significantly improved SVM-based system's performance. It was also true for the NAP-based compensation. For short authentication segments, systems without NAP was better than systems used full length available utterances for estimating the NAP directions. It was found that NAP compensation is most effective when the nuisance directions were estimated from utterances containing an amount of speech matching the authentication speech segments. The same phenomena was observed for the score normalization. It was shown that score normalization like T-Norm to be most effective when cohort speakers' utterances were matched to the utterances used for the enrollment and authentication.

One reason for the degradation of the performance of ASVS using short utterances could be that adaptation techniques fail to provide robust parameters estimation. In [96, 100, 129, 34], eigenvoice modeling proposed by Kuhn et al. for speech recognition [88], was used for adaptation arguing that the number of free parameters is small in eigenvoice modeling which suits for short utterances. In eigenvoice modeling also known as subspace adaptation (SA), a speaker-specific model can be expressed as:

$$\boldsymbol{\mu}_s = \boldsymbol{\mu} + \mathbf{V}\mathbf{y} \tag{3.53}$$

where $\mathbf{V} = \{\mathbf{v}_r\}_{r=1}^{R}$ is the R eigenvectors corresponding to the R largest eigenvalues, $\mathbf{y}$ is the parameter of speaker s resides within the speaker subspace built by $\mathbf{V}$. In [96] Lucey and Chen proposed probabilistic subspace adaptation (PSA) by considering $\mathbf{y}$ as a random variable and optimizing it

according to MAP criterion. Comparing with relevance MAP, they showed that PSA improved the performance of an ASVS when the amount of training data is limited. In [130, 129], Vogt et al. combined SA with relevance MAP adaptation.

$$\boldsymbol{\mu}_s = \boldsymbol{\mu} + \mathbf{Vy} + \mathbf{Dz}, \tag{3.54}$$

where **Dz** models residual variability that is not captured by speaker subspace. They also proposed a method similar to JFA by combining SA, relevance MAP and session variability modeling. They empirically showed that independent estimation of speaker and session subspaces performed better than their simultaneous optimization. It was also shown that for training the speaker subspace as much data as possible should be used. On the other hand, for training session subspace, it is important to match conditions and utterance lengths of data used for training subspace with the conditions and utterance lengths of the data expected to be encountered in the enrollment and authentication phases. In [34], $\boldsymbol{\mu}_s$ is calculated as follows:

$$\boldsymbol{\mu}_s = \boldsymbol{\mu} + \mathbf{Vy}(\boldsymbol{\mu}_r - \boldsymbol{\mu}), \tag{3.55}$$

where $\boldsymbol{\mu}_r$ is the speaker mean supervector derived from relevance MAP adaptation. In [35, 34], Fauve et al. also highlighted the importance of a well-tuned speech detection front-end for short-utterances. They empirically showed that a front-end that is optimized on a longer duration task generates suboptimal results when for shorter duration task. The reverse case is not so severe. Using enrollment data in the range of 2-32 seconds, Mak et al. [100] showed a comparative figure about the effect of four adaptation techniques: MAP [45], maximum likelihood linear regression (MLLR) [92], reference speaker weighting (RSW) [62] and kernel eigenspace-based MLLR (KEMLLR) [98, 67]. Using a GMM-UBM based ASVS, it was shown that KEMLLER outperforms other adaptation techniques for enrollment data between 2 to 4 seconds. KEMLLR is a kernel version of the eigenspace-based MLLR (EMLLR) adaptation [72]. EMLLR derives a small set of eigenmatrices using principal component analysis (PCA) and performs MLLR transformation by combining eigenmatrices linearly. KEMLLR finds eigenmatrices in the kernel-induced high dimensional feature space. The main shortcomings of eigenvoice modeling is that it demands huge amount of background data for training projection matrix or factor loading matrix.

The effect of using short utterances both for evaluation and development in the PLDA based ASVS was investigated in [77]. It was found that the heavy tailed PLDA (HTPLDA) performed better than the Gaussian PLDA (GPLDA) when evaluation utterance lengths were decreased. The importance of matching durations for score normalization and PLDA modeling to the expected evaluation conditions was highlighted. It was also pointed out that a pooled total-variability approach to PLDA modeling can achieve better performance than the traditional concatenated total-variability approach for short utterances in mismatched evaluation conditions.

In [76], a comparison of JFA and i-vector based systems including various compensation techniques such as WCCN, LDA, scatter difference NAP (SDNAP) and GPLDA, was done. Systems using around 2.5 minutes enrollment data were better than systems using shorter enrollment data. Marginally better performance was obtained for GPLDA based i-vector system comparing to other compensation techniques when authentication utterances were above or equal to 10 seconds. Overall, performances of all systems declined sharply once utterance lengths fall below 10 seconds. Based on these findings, appropriate session compensation techniques using short utterances are studied in [74, 75].

In [74] Kanagasundaram et al. introduced two different types of source and utterance-duration normalized LDA approaches, named SUN-LDA-pooled and SUN-LDA-concat, to compensate the session variability in i-vector based ASVS using short utterance. By capturing the source variation information from short- and full-length development i-vectors, the these approaches outperformed the traditional LDA approach. In this approach, at first between-class-scatter matrices for telephone and microphone are estimated as follows:

$$\mathrm{S}_b^{\mathrm{t}} = \alpha_{\mathrm{tf}}\mathrm{S}_b^{\mathrm{tf}} + \alpha_{\mathrm{ts}}\mathrm{S}_b^{\mathrm{ts}} \tag{3.56}$$

$$\mathrm{S}_b^{\mathrm{m}} = \alpha_{\mathrm{mf}}\mathrm{S}_b^{\mathrm{mf}} + \alpha_{\mathrm{ms}}\mathrm{S}_b^{\mathrm{ms}} \tag{3.57}$$

where α_{tf}, α_{ts}, α_{mf} and α_{ms} are respectively weighting coefficients of telephone and microphone sourced full- and short length between-class scatter estimations, $\mathrm{S}_b^{\mathrm{tf}}$ and $\mathrm{S}_b^{\mathrm{mf}}$ are individually estimated from telephone and microphone sourced full-length utterances using the following Eq. 3.58. $\mathrm{S}_b^{\mathrm{tf}}$ and $\mathrm{S}_b^{\mathrm{ms}}$ are estimated using telephone and microphone sourced short-length

utterances respectively.

$$\mathrm{S}_b = \sum_{s=1}^{S} n_s(\bar{\omega}_s - \bar{\omega})(\bar{\omega}_s - \bar{\omega})' \tag{3.58}$$

where S is the total number of speakers and n_s is number of utterances of speaker s and N is the total number of sessions. The mean i-vectors are $\bar{\omega}_s$ for each speaker and $\bar{\omega}$ for the across all speakers are defined by:

$$\bar{\omega}_s = \frac{1}{n_s}\sum_{i=1}^{n_s}\omega_i^s \tag{3.59}$$

$$\bar{\omega} = \frac{1}{N}\sum_{s=1}^{S}\sum_{i=1}^{n_s}\omega_i^s \tag{3.60}$$

The within-class scatter matrix S_w is calculated using only full-length utterances as follows:

$$\mathrm{S}_w = \sum_{s=1}^{S}\sum_{i=1}^{n_s}(\omega_i^s - \bar{\omega}_s)(\omega_i^s - \bar{\omega}_s)' \tag{3.61}$$

The LDA matrix, $\mathbf{A}$, is calculated through the eigenvalue decomposition of $\mathrm{S}_b\boldsymbol{v} = \lambda\mathrm{S}_w\boldsymbol{v}$. That means $\mathbf{A}$ is formed as the subset of eigenvectors, $\boldsymbol{v}$, having the largest eigenvalues, λ. For the SUNLDA-pooled case, $\boldsymbol{v}$ is defined by

$$(\mathrm{S}_b^t + \mathrm{S}_b^m)\boldsymbol{v} = \lambda\mathrm{S}_w\boldsymbol{v} \tag{3.62}$$

For the SUN-LDA-concat case, the $\mathbf{A}$ is formed by concatenating the telephone and microphone sourced LDA matrices, $\mathbf{A}^t$ and $\mathbf{A}^m$, as follows,

$$\mathbf{A} = [\mathbf{A}^t\mathbf{A}^m] \tag{3.63}$$

where $\mathbf{A}^t$ and $\mathbf{A}^m$ are estimated as eigenvalue decomposition of,

$$\mathrm{S}_b^t\boldsymbol{v} = \lambda\mathrm{S}_w\boldsymbol{v} \tag{3.64}$$

$$\mathrm{S}_b^m\boldsymbol{v} = \lambda\mathrm{S}_w\boldsymbol{v} \tag{3.65}$$

In [75] the short utterance variance normalisation (SUVN) and short utterance variance (SUV) modelling were introduced for PLDA based ASVS to compensate the session and utterance variations in short utterances. In these approaches, short utterance variance (SUV) matrix is estimated:

$$\mathbf{S}_{\text{SUV}} = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{\omega}_n^{full} - \boldsymbol{\omega}_n^{short})(\boldsymbol{\omega}_n^{full} - \boldsymbol{\omega}_n^{short})', \tag{3.66}$$

where i-vectors $\boldsymbol{\omega}_n^{full}$ were extracted from 100 seconds long utterances and $\boldsymbol{\omega}_n^{short}$ were extracted from 30 seconds utterances.

3.3 Robust Background Model

A background model plays an important role in dealing with both inter-session variability and short utterances. In the model based inter-session variability compensation techniques like NAP, WCCN, JFA etc., background models (e.g., NAP matrix **P**, within-class covariance matrix **W**, eigenvoice matrix **V**, eigenchannel matrix **U**) provide information about the directions of inter- and intra-speaker variations which help to build speaker-specific models excluding information unrelated to the speaker identity. For the short-utterance case, background models like UBM, eigenvoice matrix etc., provide prior distributions for the parameters of robust speaker-specific models. The standard approach is to train one background model for all target speakers in the enrollment set, $\mathcal{E}$, using all available data by matching one or two criteria. In this approach, the existence of variations in the set of target speakers and the existence of irrelevant and noisy data in the training dataset (denoted by $\mathcal{B}$) of the background models are generally ignored. Few researches on background modeling have dealt with the following two issues, even though it has been empirically proved those two issues have positive impact on the performance of ASVS.

1. robustness against enrollment data variants
2. robustness against irrelevant training data

These two issues are described briefly in the following two sessions.

3.3.1 Robustness Against Enrollment Data Variants

The homogeneity of target speakers in $\mathcal{E}$ based on some characteristics such as gender (male/female), duration of speech segments (long utterance/short utterance), transmission channel (telephone/microphone) etc., has been generally maintained in an evaluation set so that researchers can focus on the effect of a specific issue. The standard approach is to match $\mathcal{B}$ with the obvious characteristics based on which speakers included in $\mathcal{E}$ are homogeneous and train only one background model for each characteristic. For example, common approach is to train gender-dependent UBMs, i.e., one UBM for male speakers trained by using only male data, and one UBM for female speakers trained by using only female data. Empirically it has been shown that gender-dependent UBM outperformed gender-

independent UBM. However, in practical applications variability in $\mathcal{E}$ is normal. Even the enrollment set which is homogeneous according to one or two obvious factors can be heterogeneous according to more factors. For example, a set of telephone speech collected from female speakers can be heterogeneous according to the handset type (e.g., electret, carbon). The existence of data variants in $\mathcal{E}$ gets attention in [63, 142, 116]. In these studies, it has been concluded that it is better to use target speaker-specific or cluster in $\mathcal{E}$ specific UBMs rather than one UBM for all speakers in $\mathcal{E}$.

In [63], it has been empirically shown that the handset- and gender-dependent UBM outperformed the handset- and gender-independent UBM. Zhang et al. [142] proposed to use people's vocal tract length (VTL) normalization factor, α, to divide $\mathcal{B}$ into separate datasets, each part of $\mathcal{B}$ was then used to train a VTL-dependent UBM. Therefore, their system was multiple UBMs based rather than single UBM based. Their multiple background models (MBM) based system can be viewed as a natural extension of gender-dependent UBM systems. The steps of MBM based system are given below:

1. Extract normal feature vectors (e.g., MFCC, PLP coefficients) from all utterances in the background dataset, $\mathcal{B}$.
2. Train a UBM, Λ, using the feature vectors extracted from $\mathcal{B}$.
3. Generate warped features of all utterances in $\mathcal{B}$ using VTL normalization factor or warping factor, α, in the range of 0.88 and 1.12 (i.e., $0.88 \leq \alpha \leq 1.12$) with step-size 0.02 and the following bilinear frequency warping function in the filterbank analysis.

$$f^{\alpha} = f + \frac{f_u - f_l}{\pi} \arctan\left(\frac{(1-\alpha)\sin\theta}{1-(1-\alpha)\cos\theta}\right), \tag{3.67}$$

   where

$$\theta = \frac{f - f_l}{f_u - f_l}\pi, \tag{3.68}$$

   where f and f^{α} are the original and warped frequencies, respectively. f_u and f_l are the upper bound and lower bound of the filter, respectively.

   It means, if we choose n values for α in the range between 0.88 and 1.12, there will be n set of feature vectors for each utterance in $\mathcal{B}$.
4. Using the warped features of $\mathcal{B}$ (say, $\mathcal{O}^{\alpha}$), train a warped UBM, Λ^{*}

(a) Search the best warping factor in a limited grid.

$$\alpha^* = \underset{0.88 \leq \alpha \leq 1.12}{\arg\max}\ p(\mathcal{O}^{\alpha}|\Lambda) \tag{3.69}$$

(b) Update model parameters

$$\Lambda^* = \underset{\Lambda}{\arg\max}\, p(\mathcal{O}^{\alpha^*}|\Lambda) \tag{3.70}$$

(c) Set $\alpha = \alpha^*$ and $\Lambda = \Lambda^*$ and go to Step-4a.
If there is no significant difference in the warping factors between two consecutive iterations, stop parameters updating of Λ^*.

5. Estimate the best warping factor, α_r^* for each utterance, r, in $\mathcal{B}$, using warped features of that utterance (say, $\mathcal{O}_r^{\alpha}$) and warped UBM, Λ^*,

$$\alpha_r^* = \underset{0.88 \leq \alpha \leq 1.12}{\arg\max}\ p(\mathcal{O}_r^{\alpha}|\Lambda^*) \tag{3.71}$$

6. Divide $\mathcal{B}$ into N disjoint datasets, $\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_N$, according to the warping factors.
7. Train N VTL-dependent UBMs using $\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_N$.
8. Extract normal feature vectors (e.g., MFCC, PLP coefficients) from all utterances in the enrollment dataset, $\mathcal{E}$.
9. Train N speaker-specific GMMs for each speaker, S, in $\mathcal{E}$, by MAP adaptation of N UBMs using S's feature vectors.
10. Verify each authentication utterance against all the N (speaker-specific GMM, VTL-dependent UBM) pairs.
11. Fuse the scores of N GMM-UBM systems in order to get the final score.

They conducted experiments on the NIST SRE 2006 (SRE06) core task (1conv4w-1conv4w) and cross-channel task (1conv4w-1convmic). They trained eight VTL-dependent GMM-UBM systems (i.e., $N = 8$). Using gender-dependent systems and *minimum likelihood ratio* (MLR) score fusion technique, for *1conv4w-1conv4w* task they were able to reduce EER from 9.69% to 9.40% for female and from 8.38% to 8.36% male. C^{min} was reduced from 0.0449 to 0.0414 for female and from 0.0397 to 0.0371 for male. For *1conv4w-1convmic* task, EER was reduced from 11.65% to 10.76% for female and from 10.01% to 9.38% for male. C^{min} was reduced from 0.0563 to 0.0543 for female and from 0.0442 to 0.0408 for male.

Sarker et al., also proposed to train multiple UBMs using VTLN factor in [116]. The main difference between their approach and the previous

approach is that they clustered target speakers in $\mathcal{E}$, where the previous approach clustered utterances in $\mathcal{B}$. The steps of their approach are given below:

1. Extract normal feature vectors (e.g., MFCC, PLPCC) from all utterances in the background dataset, $\mathcal{B}$.
2. Train a speaker-independent UBM, Λ, using the feature vectors extracted from $\mathcal{B}$.
3. Cluster target speakers in the enrollment set $\mathcal{E}$ using VTLN factor into M disjoint clusters, $C_1, C_2, ..., C_M$.
   (a) Extract normal feature vectors (e.g., MFCC, PLPCC) from all utterances in $\mathcal{E}$.
   (b) Generate warped features of all utterances in $\mathcal{E}$ using $0.80 \leq \alpha \leq 1.20$ with step-size 0.02.
   (c) Estimate the best warping factor, α_S^* for each speaker, S, using warped features of that speaker (say, $\mathcal{Q}_S^\alpha$)

$$\alpha_S^* = \underset{0.80 \leq \alpha \leq 1.20}{\arg\max}\ p(\mathcal{Q}_S^\alpha | \Lambda) \tag{3.72}$$

4. Train speaker cluster-depndent UBMs, SC-UBMs.
   (a) Load the feature vectors of all the training utterances from all the target speakers in cluster C_j, where $j = 1, 2, ..., M$.
   (b) Build a SC-UBM^{C_j} for speaker cluster C_j with single iteration of MLLR adaptation from speaker-independent UBM.
   (c) Repeat Step-4a to -4b for all clusters.
5. Build speaker-specific GMMs from the corresponding SC-UBM using 2-iterations of MAP adaptation.
6. Verify each authentication utterance against speaker-specific GMM and SC-UBM pairs.

By using 14 clusters (i.e., $M = 14$), Sarker et al., reduced EER from 15.07% to 13.96% and C^{min} from 0.0597 to 0.0593. The EERs of both approaches are relatively high compared with other more powerful GMM-UBM systems [10]. It is because in the above two approaches complicated inter-session variability compensation techniques were not used.

3.3.2 Robustness Against Irrelevant Data

More data is not always better. Noisy, irrelevant, or weakly relevant data or outlier is *unwanted data*, since it hinders most of the data analysis like classification, regression, clustering etc., by leading to biased parameter estimation and incorrect results. Often there is no clear boundary between different kinds of unwanted data. Sometimes only distorted data as the result of an imperfect data collection, transferring or storing process, or data mixed with unexpected sources is refereed to as *noisy data*. For example, main block information in a web page along with incoherent banner advertisements, navigation bars, copyright notices, etc., is considered as noisy data [137]. According to Hawkins [61], "An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism". It is referred to as an *anomaly* in the data mining and statistics literature. Some authors named anomalies as strong outliers while noisy or erroneous data as weak outliers [86, 1]. Sometimes some data are neither noisy nor outliers, but irrelevant or weakly relevant to a particular data analysis because those data do not contain any information helpful for data analysis. For example, speech data is completely irrelevant for image recognition and male speech data is weakly relevant for female speakers.

No matter whether data is a weak outlier or strong outlier; weakly relevant or completely irrelevant, it is important to identify unwanted data and then remove it completely. If that is not possible or that causes reduction of huge amount of training data which can hinder system's performance, then transforming noisy data to useful ones is an important task. There are many approaches to do that, for example denoising techniques such as wavelet transform, different kinds of linear and non-linear filters etc., are used to repair distorted data. Different kinds of feature reduction approaches such as independent component analysis (ICA), LDA, PCA etc., have been proposed for reducing features which do not contain speaker discriminative information. Reducing irrelevant data points to make $\mathcal{B}$ smaller without hampering system's performance have been addressed in [124, 102, 101, 57, 56, 68, 125, 94].

Approaches for UBM

Huang et al. proposed to use maximum entropy in order to reduce redundant feature frames while keeping all speakers for training UBM in [68]. Let U be the unique speakers for training UBM. For each speaker, S (i.e., $S \in U$), assume that there are N_S speech files. Then the steps of their proposed method are given below:

1. Extract T_S feature frames from N_S speech files.
2. If $T_S \leq \theta$
   (a) Keep T_S frames for training UBM.

   Else
   (a) Train a speaker-specific GMM, Λ_S, using T_S feature frames . The number of Gaussian mixtures, M, is small for the robust estimation of Λ_S.
   (b) Make n_S blocks of F feature frames where $n_S = \frac{T_S}{F}$. Let each block be X_i, where $i = 1, 2, .., n_S$.
   (c) Estimate entropy of X_i

$$H(X_i) = E[I(x_f)] = \sum_{f=1}^{F} p(x_f|\Lambda_S) \log \frac{1}{p(x_f|\Lambda_S)} \tag{3.73}$$

   where

$$p(x_f|\Lambda_S) = \sum_{m=1}^{M} w_m p(x_f|\mu_m, \sigma_m) \tag{3.74}$$

   (d) Sort $H(X_1), H(X_2), ., H(X_{n_S})$ in descending order.
   (e) Choose q blocks with higher entropy so that $q \times F \leq T_S \times \alpha$ where α is a compressed factor.

Experiments were conducted by using GMM-UBM system. Setting $M = 16, B = 16$ and $\theta = 4500$, 13.68% feature frame reduction was obtained which reduced 4.69% EER and 6% DCF in the NIST SRE 2008 core task (telephone condition) having English trials only.

Sub-sampling of feature frames (SSFF) is one kind of data selection method for UBM where instead of using all available feature frames, some frames are selected for UBM training. Intelligent feature selection (IFS) method proposed by Hasan et al. [56], is a one kind of SSFF methods. It is different from other SSFF methods such as lead feature selection (LFS),

random feature selection (RFS) and uniform feature selection (UFS), since these methods do not consider any specific distribution of phonetic content over time. On the other hand, IFS method measures the similarity of successive frames using a phonetically motivated distance measure. It selects a feature frame only if that frame's dissimilarity from the most recently selected feature frame is higher than some threshold. In this approach, it is assumed that the K dimensional feature vectors of $\mathcal{B}$, originating from a specific phone, can be modeled by an independent, wide sense stationary (WSS), white Gaussian random sequence $\mathbf{X}[n]$ with a co-variance matrix,

$$\Sigma_{XX}[m,n] = \text{diag}(\lambda_1, \lambda_2, ..., \lambda_K)\delta[m-n], \tag{3.75}$$

where m, n denote feature vector indexes, and $\lambda_i = 1, 2, ..., K$ are the variances of the individual cepstral coefficients. It can be shown that the Euclidean distance, d, between two feature vectors follow a Chi-squared distribution:

$$f(d) = \frac{2^{1-K}}{\Gamma(K/2)} \frac{d^{K-1}}{\hat{\lambda}^{K/2}} \exp(-\frac{d^2}{4\hat{\lambda}}), \tag{3.76}$$

where mean, μ_d, and variance, λ_d, can be defined as:

$$\mu_d = \frac{2\sqrt{\hat{\lambda}}\Gamma(\frac{1+K}{2})}{\Gamma(K/2)}, \tag{3.77}$$

$$\sigma_d^2 = 2K\hat{\lambda} - \mu_d^2, \tag{3.78}$$

where $\hat{\lambda}$ is defined as the average variance given by

$$\hat{\lambda} = \frac{1}{K}\sum_{i=1}^{K}\lambda_i. \tag{3.79}$$

The steps of IFS are given below:

1. Extract K-dimensional N feature vectors, $\mathbf{X}$, from all speech files of $\mathcal{B}$.
2. Arrange N feature vectors as a sequence vectors, $\mathbf{X}[1], \mathbf{X}[2], ..., \mathbf{X}[N]$.
3. Set $i = 1$.
4. Consider the 1st feature vector, $\mathbf{X}[1]$, as the latest selected sequence vector, $\mathbf{S}[i]$, with mean and variance $\mu_{\mathbf{S}}[i] = \mathbf{S}[i]$ and $\lambda_{\mathbf{S}}[i] = 0$, respectively.
5. For $j = 2, 3, ..., N$

(a) Estimate mean and variance of i number of selected feature vectors $\mathbf{S}[1], ..., \mathbf{S}[i]$ and j-th sequence vector, **X**[j], using a recursive method similar to [12] as:

$$\mu_{\mathbf{S}}[i+1] = \beta_m \mu_{\mathbf{S}}[i] + (1-\beta_m)\mathbf{X}[j] \tag{3.80}$$

$$\lambda_{\mathbf{S}}[i+1] = \beta_v \lambda_{\mathbf{S}}[i] + (1-\beta_v)||\mathbf{X}[j] - \mu_{\mathbf{S}}[i+1]|| \tag{3.81}$$

(b) Estimate $\hat{\lambda}$ by setting $\lambda_{\mathbf{S}}[i+1]$ in Eq. 3.79.

(c) Estimate μ_d and λ_d by setting $\hat{\lambda}$ in Eq. 3.77 and Eq. 3.78, respectively.

(d) Estimate the distance threshold, d_{th}, for the distance between any two consecutive feature vectors

$$d_{th} = \mu_d + \sqrt{2}\sigma_d \text{erfc}^{-1}(2\alpha) \tag{3.82}$$

where erfc^{-1} is the inverse of the complementary error function (erfc) which can be defined as:

$$\text{erfc}(\alpha) = \frac{2}{\sqrt{\pi}} \int_{\alpha}^{\infty} e^{-t^2} dt \tag{3.83}$$

(e) Estimate distance between two consecutive feature vectors

$$d(i,j) = ||\mathbf{X}[j] - \mathbf{S}[i]||^{1/2} \tag{3.84}$$

(f) If $d(i,j) > d_{th}$

i. Set $i = i+1$.

ii. Select j-th feature vector, $\mathbf{X}[j]$ as the new i-th selected feature vector.

(g) $j = j+1$ and go to Step-5a.

(h) Train a UBM using $\mathbf{S}[1], \mathbf{S}[2], ..., \mathbf{X}[i-1]$

Hasan et al., conducted experiments on GMM-UBM ASVS without any inter-session variability compensation techniques. Using a $\mathcal{B}$ containing 2019 utterances (168.25 hours long in total) from 126 male speakers and setting $\alpha = 0.1, \beta_m = 0.8$ and $\beta_v = 0.8$, they achieved 99% reduction in data size and 3.8% reduction in EER for the NIST SRE 2008 (tel-tel) male task.

Approaches for SVM

A GMM-SVM system proposed by Campbell et al. [18] is accepted as one of the state-of-art ASVS for text-independent speaker verification.The training of a target speaker-specific SVM involves positioning a hyperplane in a high dimensional space which maximizes the margin between target speaker class, $\mathcal{T}$, and impostor class, $\mathcal{I}$. The position of the separating hyperplane is defined by a subset of members of $\mathcal{T}$ and $\mathcal{I}$ known as *support vectors*. Support vectors of an SVM hyperplane normal $w = \sum_i x_i y_i \alpha_i$ are the training vectors, x_i with a class label $y_i \in \{-1, 1\}$ and coefficient $\alpha_i > 0$. The members of $\mathcal{T}$ and $\mathcal{I}$ which are not selected as support vectors in the training phase of an SVM do not have any contribution in the authentication phase of that SVM (see Section 2.4.3).

The standard approach is to use the same background dataset, $\mathcal{B}$ for modeling $\mathcal{I}$ for all $\mathcal{T}$ in an enrollment set $\mathcal{E}$. It has been shown that effective selection of $\mathcal{B}$ for $\mathcal{I}$, which includes as much data as possible which are close to the $\mathcal{E}$, can improve performance of the SVM-based ASVS, while verification accuracy degrades when whole $\mathcal{B}$ is used as $\mathcal{I}$.

McLaren et al., proposed to select a subset of $\mathcal{B}$ based on the assumption that more frequently selected data in $\mathcal{B}$ as support vectors belong to $\mathcal{I}$ are likely to be more important than the rarely selected data in $\mathcal{B}$ as support vectors belong to $\mathcal{I}$ [102, 101]. Let the enrollment set, $\mathcal{E}$ of a development set, $\mathcal{D}$, has n number of target speakers. The steps of their method is given below:

1. Train n speaker-specific SVMs using the same $\mathcal{B}$ for $\mathcal{I}$ in every SVM.
2. Calculate the *support vector frequency* (SVF) for each vector, j, in $\mathcal{B}$

$$\text{SVF}_j = \sum_{i=1}^{n} \phi(\alpha_j^i), \tag{3.85}$$

   where,

$$\phi(\alpha) = \begin{cases} 1 & \text{if } \alpha > 0 \\ 0 & \text{if } \alpha = 0 \end{cases} \tag{3.86}$$

3. Choose k vectors from $\mathcal{B}$ with the highest SVF and make refined background dataset, $\mathcal{R}_k \subset \mathcal{B}$.
4. Tune k on $\mathcal{D}$ to find $\mathcal{R}_k$ which maximizes SVM's verification performance.

5. Use $\mathcal{R}_k$ as the background dataset for impostor class $\mathcal{I}$ of the evaluation set, $\mathcal{V}$.

In that study the support vector frequencies (SVFs) of a large set of impostor examples was calculated for evaluating the NIST SRE 2006 SRE core (English only) task. Evaluation showed that a refined background dataset, $\mathcal{R}_k$, of the 500 highest-ranking observations maximized performance, giving relative improvements of up to **22%** in both minimum DCF and EER over the complete $\mathcal{B}$.

The support vector frequency (SVF) method selects N vectors for $\mathcal{I}$ by conducting experiment on a development set, $\mathcal{D}$. However, the fixed number of vectors for $\mathcal{I}$ does not always provide consistent performance in different evaluation data, $\mathcal{V}$. In [125] Suh et al., showed that for the NIST SRE 2008 male core task, SVF based method found $R_{k=500}$ as the optimum subset of $\mathcal{B}$ consisting of the NIST SRE 2004 and NIST SRE 2005. However, for the NIST SRE 2010 male core task, $R_{k=500} \subset \mathcal{B}$ did not produce the best result. The size of the optimum subset for this task was 300. This variation in performance across the datasets suggests that the SVF based data selection method needs to be improved. Suh et al. proposed steps in order to improve SVF based dataset selection for impostor models:

1. Train speaker-specific SVMs for all enrollment speakers in $\mathcal{V}$ using $\mathcal{B}$.
2. Estimate SVF of each vector in $\mathcal{B}$ using Eq. 3.85.
3. Order vectors in $\mathcal{B}$ based on SVF value.
4. Separate ordered vectors of $\mathcal{B}$ into three sets, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$. Use $\mathcal{B}_1$ of size a for measuring decision threshold θ, $\mathcal{B}_2$ of size b for finding $\mathcal{R}$ and $\mathcal{B}_3$ of size c for error estimation.
5. Make $l+1$ overlapped datasets of size p from $\mathcal{B}_2$ (i.e., $\mathcal{B}_2^p$) by increasing p in increments of 100 (i.e., $p = 100,\ p+1 = 200, ..., p+l = 100 \times l$).
6. For each enrollment speaker, n, in $\mathcal{V}$, do:
   (a) Train speaker-specific SVM using $\mathcal{B}_2^p$ for impostor class $\mathcal{I}$.
   (b) Measure error of n's SVM using $\mathcal{B}_3$

$$\text{Err}_{n,p} = \frac{1}{c}\text{card}\{j : O_{p,j} - \theta < 0), \tag{3.87}$$

   where 'card{}' is the cardinality and $O_{p,j}$ is the output function of j-th vector, x_j, in $\mathcal{B}_3$. $O_{p,j}$ is defined as:

$$O_{p,j} = w_p \cdot x_j - b, \tag{3.88}$$

where w_p is the norm vector and b is the bias of n's SVM. Decision threshold θ is defined as:

$$\theta = a \sum_{j=1}^{a} O_{p,j}. \tag{3.89}$$

(c) Calculate *error difference* (ED) as:

$$\text{ED}_{n,p} = \text{Err}_{n,p} - \text{Err}_{n,p+1} \tag{3.90}$$

(d) Select the steepest slope in ED. The simplified method to find the steepest slope for background dataset is

$$p* = \arg\max_{p}(\text{ED}_{n,p} - \text{ED}_{n,p+1}) \tag{3.91}$$

(e) Choose $\mathcal{B}_2^{p*}$ as the refined dataset $\mathcal{R}$ for enrollment speaker n.

In general there is an unlimited available database for $\mathcal{I}$ (i.e., $\mathcal{B}$ is huge) whereas the data for $\mathcal{T}$ is limited. For example, for the core task of the NIST SRE 2004-2010, the ratio between the members of $\mathcal{T}$ and $\mathcal{I}$ could be 1:1300. In [94], Liu et al., proposed a balanced impostor selection method with respect to the entire enrollment speakers' space.

1. Train a universal SVM (noted as u-SVM) by using all the enrollment speakers' vectors in $\mathcal{E}$ as positive examples and all impostors' vectors in $\mathcal{B}$ as negative examples.
2. Estimate SVF of each vector in $\mathcal{B}$ using Eq. 3.85.
3. Order vectors in $\mathcal{B}$ based on SVF value.
4. For j-th enrollment speaker, where $1 \leq j \leq S$ and S is the total number of enrollment speakers in $\mathcal{E}$, do:
   (a) Train j-th speaker-specific SVM (notes as j-SVM) using j-speaker's vectors as positive examples and the negative support vectors of the u-SVM as negative examples.
   (b) Pool negative support vectors of j-SVM and p most frequent negative support vectors in $\mathcal{B}$ to build a set of vectors, $\mathcal{R}_j$, for negative class of j-th speaker. Decide p as proposed in [125].
   (c) Train j-th speaker-specific SVM using j-speaker's vectors as positive examples and $\mathcal{R}_j$ as negative examples.

Evaluations were done on 5min-5min telephone-telephone core-condition of the NIST SRE 2008 (SRE08) and 2010 (SRE10) corpora. $\mathcal{V}$ contained only male speakers. EER was reduced from 6.01% to 4.90% for SRE08 and from 6.14% to 5.67% for SRE10.

Approaches for T-Speakers in T-Normalization

The standard approach is to match some broad speaker-specific information, such as speaker's gender or telephone set and draw T-norm speakers from the same background dataset used for training UBMs. There has been little research in more speaker-specific, data driven approaches for selecting T-norm speakers. Sturim et al., proposed a data driven approach for target speaker-specific T-norm speaker selection in order to improve verification performance in [124]. They named their approach *Adaptive-Tnorm* or *ATnorm*. Steps of their proposed method are given below:

1. Pool utterances of P speakers from a background dataset, $\mathcal{B}$, containing utterances varying in handset types, durations, number of sessions, age and many other factors.
2. Train P GMM-UBMs called as *ATnorm models* for P number of Tnorm speakers.
3. Choose N impostor authentication speech segments from $\mathcal{B}$ not included in the training data of P Atnorm models.
4. Verify N utterances against P ATnorm models and generate P number of score vectors, $L_p \in \mathbb{R}^N$, where $p = 1, 2, ..., P$. In L_p each dimension contains LLR score of one ATnorm model.
5. For each target speaker, s, in the enrollment set $\mathcal{E}$ of a development set, $\mathcal{D}$:
   (a) Train a speaker-specific GMM-UBM
   (b) Verify N utterances against s's model and generate N-dimensional score vector, $L_s \in \mathbb{R}^N$.
   (c) Estimate city block distance between L_s and L_p

$$d(s,p) = \sum_{i=1}^{N} |L_s(i) - L_p(i)|, \text{ where p=1, 2, ..., P.} \qquad (3.92)$$

   (d) Choose K nearest ATnorm models of the speaker s according to the distance between L_s and L_p.

(e) Tune K

6. For each target speaker in the enrollment set $\mathcal{E}$ of an evaluation set, $\mathcal{V}$, do Steps 5a - 5d using K tuned on $\mathcal{D}$.

By using $P_{\text{male}} = 435, P_{\text{female}} = 550$ and $N = 800$, they tuned K to 55. They conducted experiment on the NIST SRE 2004 extended task (8conv-1conv) and showed that ATnorm outperformed traditional T-norm.

McLaren et al., showed that support vector frequency (SVF) based approach used for selecting data for the negative class in SVM based system (see Section 3.3.2) can also be used for finding suitable T-norm speakers both for JFA and SVM based ASVS [102]. They were able to drop EER from 3.20% to 2.93% and C^{min} from 0.0171 to 0.0137 for the NIST SRE 2006 1sided English Task [105] using a JFA based ASVS. Using a SVM ASVS with NAP, they dropped EER from 4.93% to 4.48% and C^{min} from 0.0230 to 0.0202.

Summary of Previous Works

The summary of some previous works on reducing irrelevant data points from $\mathcal{B}$ in speaker verification is given in Table 3.1.

Approaches for PLDA Model

Recently, PLDA model has become one of the state-of-art methods for inter-session variability compensation in an i-vector based ASVS (See Sections 2.4.4 and 3.1.3). Although data-driven approaches have been proposed for reducing irrelevant data points in $\mathcal{B}$ for UBM and SVM, it was not done for PLDA model. In order to train the parameters of a PLDA model, multi-session recordings from several hundred speakers, resulting in several thousands of recordings from multiple databases, are typically used. For example, research groups involved in the NIST speaker recognition evaluation (SRE) typically use utterances from all NIST 2004-2005 data along with the Switchboard II, Phases 1, 2 and 3; Switchboard Cellular, Parts 1 and 2 data, and Fisher data. However, there is no evidence that using all the available data would guarantee the best PLDA model.

Based on the experiences from the other models such as UBM, SVM or JFA, researchers typically use gender-dependent PLDA models. Senous-

Table 3.1: Previous work on Background models. First line of 'Result' column shows EER (%) and second line shows $100 \times C^{min}$. Left side of '–>' is baseline performance and right side is proposed method performance.

Researchers	Background Model	Proposed Method	Evaluation Dataset	Result
Zhang et al. [142]	UBM	Clustering $\mathcal{B}$ based on VTLN factor and training multiple UBMs	NIST SRE 2006 1conv4w-1convmic (female)	11.65 –>10.76 5.63 –>5.43
Sarkar et al. [116]	UBM	Clustering $\mathcal{E}$ based on VTLN factor and training cluster-dependent UBMs	NIST SRE 2004 core task	15.07 –>13.96 5.97 –>5.93
Huang et al. [68]	UBM	Feature frames reduction from $\mathcal{B}$ based on maximum entropy	NIST SRE 2008 core task (tel-tel), English trials	4.05 –>3.86 1.73 –>1.63
Hasan et al. [57, 56]	UBM	Feature frames reduction from $\mathcal{B}$ based on sub-sampling of frames	NIST SRE 2008 core task (tel-tel), English trials	11.43 –>10.99 using 1% data of baseline
McLaren et al. [102, 101]	Impostor models in SVM	Choosing k impostors having high support vector frequency (SVF)	NIST SRE 2006 core task, English trials	3.21 –>2.49 1.52 –>1.18
Suh et al. [125]	Impostor models in SVM	Estimating error on the support vector for each enroll speaker	NIST SRE 2008 core task (tel-tel), male trials	6.01 –>4.98 6.56 –>5.42
Liu et al. [94]	Impostor models in SVM	Selecting universal background support impostor models	NIST SRE 2008 core task (tel-tel), male trials	4.98 –>4.90 5.42 –>4.90
Sturim et al. [124]	T-speakers	Selecting k nearest impostor models	NIST SRE 2004 extended task, (8conv-1conv)	6.84 0.27
McLaren et al. [102]	T-speakers	Selecting k impostors having high SVF	NIST SRE 2006 core task, English trials	4.93 –>4.48 2.30 –>2.02

saoui et al. [118], empirically showed that gender-dependent PLDA models outperformed gender-independent PLDA models even the later one had all

available data from male and female speakers while the former ones had only gender specific data. Obviously, a speaker's acoustic properties depend not only on gender but also on the physical properties of the vocal tract, dialect, age etc. In addition, phone sets, transmission channel types or background noises are known to greatly affect the acoustic properties of a recording. Kanagasundaram et al. [77], showed that the PLDA model trained by utterances whose lengths matched with those utterances in the evaluation set, $\mathcal{V}$, performed better than that trained by full-length utterances. In that study, speech segments in $\mathcal{B}$ were divided into smaller pieces so that their lengths were almost the same as that for $\mathcal{V}$. These studies showed that data relevancy is more important for the verification performance than data size. One could continue to experiment in this way by exploring whether it is beneficial to find a subset of $\mathcal{B}$ which is relevant to $\mathcal{V}$ with respect to other properties such as microphone, age, emotions, transmission channel or accent. However, this kind of meta-data is generally not available. Moreover, it is not always obvious which properties are important to be matched to find relevant data.

Chapter 4

Robustness Against Enrollment Data Variants

In this chapter, we describe our proposed method *acoustic forest* in details which deal with the robustness issue of background models against enrollment data variants for SMAP based ASVS. The organization of this chapter is as follows: Section 4.1 points out our motivation. Section 4.2 illustrates how to grow an acoustic forest. Section 4.3 describes our experimental setup for conducting experiments in order to evaluate acoustic forest. Section 4.4 presents results.

4.1 Motivation

As discussed in Section 2.5.2, SMAP adaptation technique provides robust parameter estimates for very limited amount of adaptation data while keeping the desirable asymptotic properties of relevance MAP. Therefore for very short utterances like 10 seconds or less, SMAP adaptation is better than relevance MAP adaptation in order to train speaker-specific GMMs from a speaker-independent GMM. The main advantage of SMAP over eigenvoice adaptation is that using hierarchical adaptation approach SMAP demands less background data.

In SMAP adaptation, a tree structure obtained by clustering Gaussians offers a convenient way to capture the hierarchical structure of the acoustic space of the human voice. Different speakers may have different structures of the acoustic space depending on factors such as their language, accents or

pronunciation particularities. For example, we can mention the differences between English and Spanish. Among consonants, there are 15 common phonemes in both languages, 5 Spanish phonemes do not occur in English, whereas 9 English phonemes do not occur in Spanish. Therefore, when a Spanish speaker speaks a English word having phoneme not in Spanish, he/she may pronounce that phoneme like a Spanish phoneme whose pronunciation is close to that English phoneme. For example, $/s/$ and $/z/$ are two different phonemes in English, but a Spanish speaker typically pronounce them as $/s/$. In Spanish, pitch does not vary as it does in English. Therefore, a non-native English speaker from Spain may sound monotone when speaking English. These kinds of differences can be seen not only between English and Spanish, but also between other two languages, even two different dialects of same languages. For example, Japanese speakers pronounce $/l/$ as $/r/$ when speak in English, Swedish or English speakers pronounce $/d^h/, /t^h/$ as $/d/, /t/$ when speak in Bengali. If $\mathcal{B}$ contains speech data from multiple languages spoken by native and non-native speakers, we may assume that the hierarchical structure of the acoustic space cannot be shared among all the speakers. That means, it is reasonable to think that the optimal tree structure differs from an English speaker to a Spanish speaker. In other words, some tree structures may be adapted more efficiently to English speakers than Spanish speakers.

As discussed in Section 3.3.1, using enrollment speaker-specific background model or sub-cluster of enrolled speakers specific background models improves systems performance. Therefore, different tree structures should be provided for different enrollment speakers. However, until now no methods for obtaining such trees automatically are known. On the other hand, to find the optimal tree structure for every speaker empirically is computationally expensive when the number of speakers is large, and demands a large amount of data. The easiest solution of this problem is to use a set of trees instead of using a single tree, assuming that each speaker will find its appropriate tree. We define the set of trees as an acoustic forest [8]. Combining decisions of multiple ASVS using trees in the acoustic forest gives robustness against enrollment data variants.

4.2 Acoustic Forest

Acoustic forest is a set of trees obtained by clustering Gaussian components of a speaker-independent UBM. It is different from the well-known random forest [11]. In a random forest each tree is a decision tree, i.e., each tree is used for the classification or regression purpose. On the other hand, each tree in our acoustic forest is used to provide prior information about the GMM parameters in a hierarchical manner for building speaker-specific GMMs. The classification or decision making task is done by speaker-specific GMMs. The steps of an acoustic forest-based ASVS are given below:

1. Train a speaker-dependent UBM using available background data, $\mathcal{B}$.
2. Decide the number of trees, N, heuristically for the acoustic forest.
3. For $T = 1, 2, ..., N$
   (a) Set the number of layers L and the number of branches $B_r^{(l)}$ from a node r at the l-th layer of T-th tree.
   (b) Construct T-th tree following the steps mentioned in Section 4.2.1.
   (c) Build speaker-specific GMMs by adapting T-th tree with speaker-specific data by following the steps described in Section 2.5.2.
   (d) Generate scores for a trial.
4. Combine scores of N trees and take final decision about a trial.

In the acoustic forest based ASVS, the most troublesome task is to decide the number of trees and the structure of each tree. Until now there is no automatic way to fix these issues. We need to depend on a development set, $\mathcal{D}$, to solve this problem. During choosing a tree structure, we need to ensure that the number of leaf-nodes are less than or equal to the number of Gaussian components.

4.2.1 Tree Construction

A tree is constructed by clustering Gaussian components of a UBM. For clustering, two widely used distance measure are the symmetric Kullback-Leibler (KL) divergence and Bhattachariyya distance [119, 136]. In this thesis, we use the former distance measure. Assuming the covariance matrices to be diagonal, the KL divergence between two Gaussian components,

$g_a(\cdot)$ and $g_b(\cdot)$, can be written as

$$d(a,b) = \sum_{i=1}^{F}[\frac{\sigma_a^2(i) - \sigma_b^2(i) + (\mu_b(i) - \mu_a(i))^2}{\sigma_b^2(i)} + \frac{\sigma_b^2(i) - \sigma_a^2(i) + (\mu_a(i) - \mu_b(i))^2}{\sigma_a^2(i)}], \tag{4.1}$$

where $\mu_a(i)$ is the i-th element of F-dimensional mean vector μ_a and $\sigma_a^2(i)$ is the i-th diagonal element of covariance matrix $\sum_a$.

The algorithm for obtaining a tree from a UBM with M Gaussians is given below:

1. Set:
   (a) k to be the root node
   (b) G_k to be a set of all the M Gaussians governed by node k,
   (c) $B_k^{(1)}$ to be the number of children of node k
   (d) l to be 1.
2. Calculate the node pdf g_k for node k using the following formulas:

$$\mu_k(i) = \frac{1}{M_k} \sum_{m \in G_k} \mu_m(i), \tag{4.2}$$

$$\sigma_k^2(i) = \frac{1}{M_k}[\sum_{m \in G_k} (\sigma_m^2(i) + \mu_m^2(i)) - M_k \mu_k^2(i)], \tag{4.3}$$

   where M_k is the number of Gaussian components included in G_k.
3. If l is equal to L, stop clustering, else go to Step 4.
4. Compute the initial pdf for n child nodes using the $minimax$ method:
   (a) Find n Gaussian components from G_k:
      i. The 1st Gaussian is $g_{c_1}(\cdot) = g_{\hat{m}}(\cdot)$ where

$$\hat{m} = \arg\max_m d(m,k). \tag{4.4}$$

      ii. The remaining $(n-1)$ Gaussians will be $g_{c_p}(\cdot) = g_{\hat{m}}(\cdot)$ where

$$\hat{m} = \arg\max_m \min_{q \in G_{ck}} d(m, c_q). \tag{4.5}$$

      Here G_{ck} is the set of Gaussians already assigned to the child nodes of node k, $1 \leq p \leq n-1$ and $1 \leq q \leq n-2$.

(b) Interpolate the node pdf of node k and the initial node pdf of each child node c_p to create a new node pdf for c_p as follows:

$$\hat{\mu}_{c_p}(i) = (1-\alpha)\mu_k(i) + \alpha\mu_{c_p}(i), \tag{4.6}$$

$$\hat{\sigma}^2_{c_p}(i) = (1-\alpha)(\sigma^2_k(i) + \mu^2_k(i)) + \alpha(\sigma^2_{c_p}(i) + \mu^2_{c_p}(i)) - \hat{\mu}_{c_p}, \tag{4.7}$$

where $0 \leq \alpha \leq 1$.

5. Repeat the following k-means procedures until the grand sum of distances, $\mathcal{GD}$, converges:
   (a) For each Gaussian component in G_k, calculate the distance from it to each child node pdf of the l-th layer by using Eq. (4.1), and assign it to the nearest child node.
   (b) Recalculate the child node pdf by using Eq. (4.2) and Eq. (4.3).
   (c) Using Eq. (4.1), calculate the sum of distances, $\mathcal{D}$, from each child node to each of its mixture components and then obtain $\mathcal{GD}$ by accumulating all $\mathcal{D}$.
6. Set each child node to be node k and its corresponding subset of Gaussian components to be G_k. Increase l and go to Step 4.

Figure 4.1 shows a schematic example of tree construction of Gaussian components of UBM in SMAP. In the acoustic forest, the number of layers and number of branches of each node will vary from tree to tree. Figure 4.2 shows the acoustic forest having six tree structures for the UBM mentioned in Fig. 4.1(a).

4.2.2 Decision Making

There are different ways to combine the decisions of multiple SMAP adapted GMM-SVM systems with different tree structures. Like the random forest algorithm [11], we can use a voting approach or we can fuse the scores of different systems and take the decision by setting a threshold on the fused score. In this thesis we use three score fusion techniques.

Let $s_1, s_2, ..., s_L$ be the L scores of a trial produced by L SMAP adapted systems. Then the fused score, $\hat{S}$ of the claimed speaker can be calculated in the following ways:

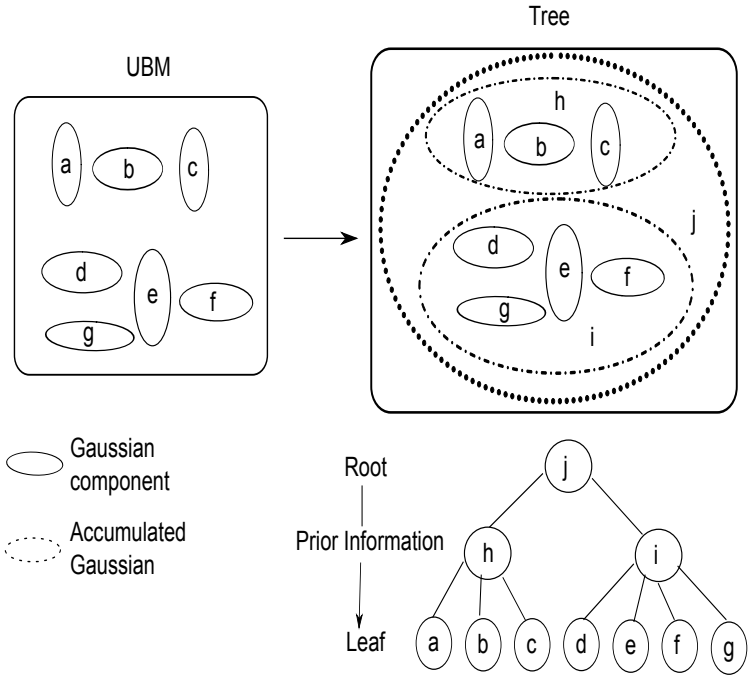

Figure 4.1: *An example of a tree structure of Gaussian components in SMAP. Each of a, b, ..., g is a Gaussian component of a UBM. h, i and j are parent Gaussians of $\{a, b, c\}$, $\{d, e, f, g\}$ and $\{h, i\}$, respectively.*

- Maximization

$$\hat{S} = \max(s_1, s_2, ..., s_L) \tag{4.8}$$

- Sum

$$\hat{S} = \sum_{l=1}^{L} s_l \tag{4.9}$$

- Multilayer Perceptron (MLP)

$$\hat{S} = \frac{1}{1 + \exp(-(\sum_{h=1}^{H} w_{h,o} y_h + \delta_{h,o}))}, \tag{4.10}$$

where H is the number of neurons in the hidden layer; $w_{h,o}$ and $\delta_{h,o}$ are the weights connecting the hidden layer and the single output neuron of an MLP; y_h is the output of the h-th neuron of the hidden layer which is obtained by the following sigmoid function:

$$y_h = \frac{1}{1 + \exp(-(\sum_{l=1}^{L} w_{l,h} s_l + \delta_{l,h}))} \tag{4.11}$$

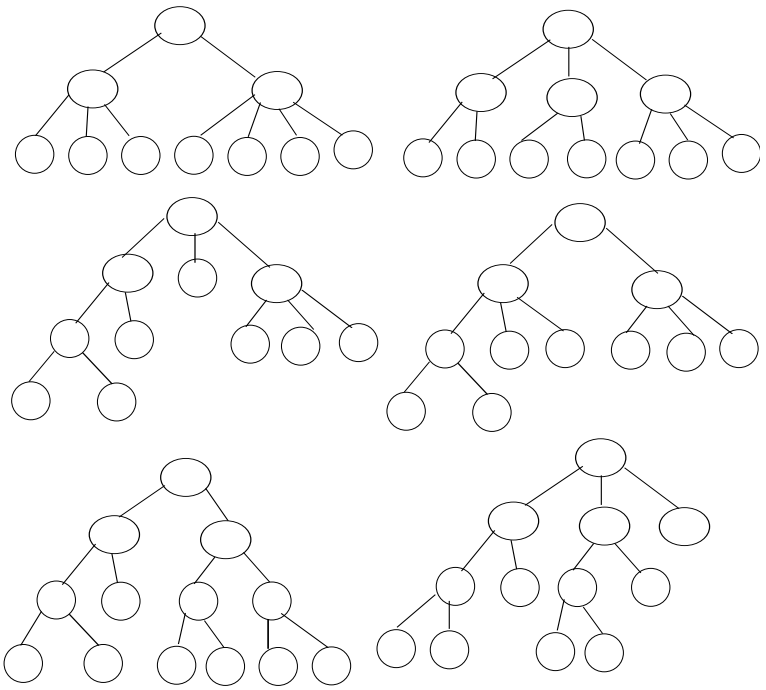

Figure 4.2: *Acoustic forest having six tree structures for the UBM with seven Gaussian components. The number of layers and the number of branches of each node vary from tree to tree.*

where $w_{l,h}$ and $\delta_{l,h}$ are the weights connecting the l-th neuron of the input layer and the h-th neuron of the hidden layer, and s_l is the score of the l-th system.

4.3 Experimental setup

In our evaluation, we used a GMM-SVM ASVS proposed by Campbell et al. [18] (See Section 2.4.3). Performance of our ASVS was measured by carrying out experiments on the 10sec4w-10sec4w task of the NIST SRE 2006 [105].

4.3.1 Development set and Evaluation set ($\mathcal{D}$ and $\mathcal{V}$)

The enrollment set, $\mathcal{E}$ and authentication set, $\mathcal{A}$, designed for the 10sec4w-10sec4w task of the NIST SRE 2005 (SRE05) [104] were used as the development set, $\mathcal{D}$, whereas $\mathcal{E}$ and $\mathcal{A}$ prepared for the 10sec4w-10sec4w task of

Table 4.1: Details of the evaluation set, SRE06. #M: the number of models in $\mathcal{E}$, #Te: the number of test files in $\mathcal{A}$, #T: the number of total trials, #Tr: the number of target trials, #Nt: the number of non-target trials.

SRE06	#M	#Te	#Tr	#Nt
Male	316	958	1319	13695
Female	415	1205	1652	16889
Total	731	2163	2971	30584

the NIST SRE 2006 (SRE06) [105] were used as the evaluation set, $\mathcal{V}$. Details of $\mathcal{V}$ are given in Table 4.1. $\mathcal{E}$ of SRE06 mostly included English speech segments and some speech segments from seven other languages: *Bengali*, *Thai*, *Hindi*, *Urdu*, *Chinook Jargon*, *Russian* and *Korean*. On the other hand, the speech segments in $\mathcal{A}$ of SRE06 were from 14 languages. The length of speech segments both in $\mathcal{E}$ and $\mathcal{A}$ was approximately 10 seconds.

4.3.2 Background Dataset ($\mathcal{B}$)

We used the NIST SRE 2004 database as the background dataset $\mathcal{B}$. This dataset has conversational telephone speech. Most of the speakers are bilingual who speak Arabic, Mandarin, Russian or Spanish in addition to English. Each speech file in $\mathcal{B}$ consists of 5-minute excerpt from a 6-minute call. Without silence, each file contains 2.5 minutes long speech on average.

4.3.3 Pre-Processing and Training Models

Regarding feature extraction, we first removed the non-speech part from the speech segments using the information in the transcript files provided by the NIST SRE organizer. We broke each speech segment into frames of 30 ms long with a frame rate of 100 frames/sec. We pre-emphasized each frame with a pre-emphasis factor of 0.97 and applied a Hamming window. We computed PLP coefficients and MFCC, augmented with the energy coefficient and first and second derivatives. Cepstral mean subtraction was applied to remove static channel effects. A three layer MLP with three hidden neurons in the hidden layer was trained for score fusion by using MATLAB.

UBM

We trained one gender-independent UBM and two gender-dependent UBMs using $\mathcal{B}$ (Table 4.2). We applied five iterations of Baum-Welch re-estimation to estimate the parameters of UBM. The feature extraction and UBM part were implemented by using the hidden Markov model toolkit (HTK) [138].

Table 4.2: UBM

UBM Type	No. of Speech Segments	No. of Frames
Gender-independent UBM	4806	66848003
Male UBM	1974	27046663
Female UBM	2832	39801340

Tree Structure

We constructed two acoustic forests with two groups of trees. In the first forest, there were eight binary trees and in the second forest there were 10 different trees having odd number of children for each node as shown in Table 4.3.

GMM-SVM

By using adaptation techniques, MAP and SMAP, speaker-specific GMMs were made from the UBM, whose means were then stacked to make supervectors. A linear SVM for each speaker was trained by using one target speaker-specific supervector and multiple impostor supervectors. For building impostor supervectors, 604 speech segments were randomly selected from 4806 speech segments used to train UBM. Among 604 impostors, 242 were male speakers and 362 were female speakers. Therefore, the number of background speakers was 242 for male GMM-SVM and 362 for female GMM-SVM. The SVM classifier was made by using LIBSVM [20].

Table 4.3: Tree Structures for SMAP. The design of a tree is written as $n_1 - n_2 - ... - n_l$ where n_l represents the maximum number of child nodes belonging to each node of the l-th layer.

Tree Structure	No. of Child Nodes
Forest-1	
1. 2-2	4
2. 2-2-2	8
3. 2-2-2-2	16
4. 2-2-2-2-2	32
5. 2-2-2-2-2-2	64
6. 2-2-2-2-2-2-2	128
7. 2-2-2-2-2-2-2-2	256
8. 2-2-2-2-2-2-2-2-2	512
Forest-2	
1. 3-3	9
2. 5-5	25
3. 7-7	49
4. 9-9	81
5. 11-11	121
6. 13-13	169
7. 15-15	225
8. 17-17	289
9. 19-19	361
10. 21-21	441

4.3.4 Tuning Parameters

For tuning parameters we conducted experiment on SRE05. First we conducted an experiment on relevance MAP-adapted GMMs with 32 Gaussian components. By setting the relevance factor equal to 10, we found that the system using the gender-dependent UBMs was better than the system using the gender-independent UBM. We also noticed that PLP outperformed MFCC. So, for further experiments, we used gender-dependent UBM and

PLP features.

For 2 or 3 minutes speech segments, the value of relevance factor, τ is generally set to the value from 10 to 20. However, decreasing τ from 10 to 1, we noticed that the lower the τ, the better the performance for 10 second test. We also noticed that $\Delta\Delta$ coefficient degraded the performance of ASVS and log-energy had a great impact for improving the system performance. The lowest EER was achieved for 32 dimensional feature vector with Δ, $\log Energy$ and $\Delta \log Energy$. For further experiments, we used 32 dimensional feature vector and $\tau = 1$.

In order to avoid computational complexity, we did not use GMMs with more than 1024 Gaussian components. We noticed that the performance of our MAP adapted system improved, when we increased the number of Gaussian components until 512. We, therefore, set the number of Gaussian components to 512.

After selecting feature vector type and setting UBM parameters, we focused on acoustic forest part. The performances of most of the binary tree adapted systems were worse than the relevance MAP adapted system. Therefore, we discarded Forest-1 for the evaluation dataset, SRE06.

4.4 Results

Table 4.4 shows the EER(%) and C^{min} of our MAP and SMAP adapted systems where we used the gender-dependent UBM with 512 Gaussian Components, 32 dimensional PLP feature vectors (i.e. 15 PLP + 15 ΔPLP + E + ΔE), and set the relevance factor to 1. Most of the SMAP-adapted systems using trees in Forest-2 outperformed the relevance MAP-adapted system. Error rates of SMAP-adapted systems consistently decreased as the number of nodes got larger. The best relative improvement for individual SMAP systems, around 3.2%, was obtained for the 21_21 tree structure-based system without T-Normalization. T-Normalization helped to drop the EER slightly for both MAP and SMAP adapted systems.

As shown in Table 4.5, score fusion techniques improved the SMAP adapted system by decreasing only EER. Two fusion techniques, sum and MLP, gave the same performance. Using acoustic forest, 1.5% relative reduction was gained in EER compared to the single tree structure-based system.

Table 4.4: *EER and C^{min} for GMM-SVM systems using MAP and SMAP adaptation on the 10sec4w-10sec4w task of 2006 NIST SRE. The design of a tree is written as n_1_n_2 where n_l represents the maximum number of child nodes belonging to each node of the l-th layer. Each leaf node corresponds one component in GMM.*

Adaptation	No Norm		T-Norm	
	EER(%)	C^{min}	EER(%)	C^{min}
MAP	**27.7**	**0.0917**	**27.4**	**0.0910**
SMAP 3_3	28.2	0.0959	27.8	0.0941
SMAP 5_5	27.9	0.0943	27.6	0.0921
SMAP 7_7	27.7	0.0943	26.9	0.0917
SMAP 9_9	27.4	0.0937	26.9	0.0918
SMAP 11_11	26.9	0.0936	**26.6**	0.0918
SMAP 13_13	27.1	0.0932	26.6	0.0913
SMAP 15_15	27.3	0.0930	26.9	**0.0909**
SMAP 17_17	27.2	0.0933	27.0	0.0910
SMAP 19_19	27.2	0.0927	26.9	0.0913
SMAP 21_21	**26.8**	**0.0922**	26.9	0.0915

Table 4.5: *Comparison of the EER and the MDC for fusion of ten SMAP adapted systems with and without T-Norm on the NIST 2006 SRE 10sec4w-10sec4w task.*

Fusion	No Norm		T-Norm	
	EER(%)	C^{min}	EER(%)	C^{min}
Maximization	27.2	0.0945	26.5	0.0915
Sum	26.5	0.0928	26.2	0.0909
MLP	26.5	0.0922	26.2	0.0908

Chapter 5

Robustness Against Irrelevant Training Data

In this chapter, we discuss about our proposed methods for reducing unwanted data in order to enhance robustness of PLDA models against irrelevant training data. The organization of this chapter is as follows: Section 5.1 describes our motivation. Section 5.2 presents a background study. Section 5.3 presents our proposed data selection methods and their variants. Section 5.4 and Section 5.5, describe our experimental setup and results, respectively. Section 5.6 includes analysis of different issues regarding the data selection schemes, as well as experiments with some of their modifications and extensions described in Section 5.3.

5.1 Motivation

As discussed in Section 3.3.2, more data is not always better. Data relevancy is more important for the verification performance than data size. Outliers, noisy and irrelevant data can hinder system's performance significantly. Therefore, they are unwanted. For enhancing system's performance, it is necessary to remove unwanted data from background dataset, $\mathcal{B}$. PLDA model is considered as one of the-state-of-the-art methods for inter-session variability compensation technique now these days. No data-driven approach has been proposed for cleaning unwanted data from $\mathcal{B}$ used for training PLDA models.

In many real applications such as on-line bank services for registered

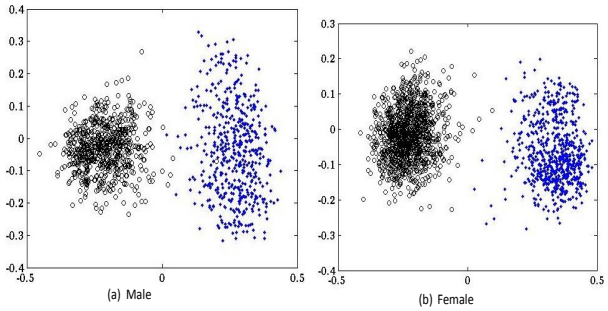

Figure 5.1: Clusters formed by i-vectors extracted from phone- and microphone-utterances after principal component analysis (PCA). x-axis is for the first principal component and y-axis is for the second principal component. Black circles represent i-vectors extracted from the phone-utterances and blue dots represent i-vectors extracted from the microphone-utterances. (a) There are 1270 male i-vectors. Among them 648 are from the phone-utterances and 622 are from the microphone-utterances. (b) There are 1993 female i-vectors. Among them 1140 are from the phone-utterances and 853 are from the microphone-utterances.

customers, we can access the enrollment set, $\mathcal{E}$, during the development phase of the system. Targeting such applications, we propose to use $\mathcal{E}$ for selecting $\mathcal{S} \subset \mathcal{B}$ that has similar properties to $\mathcal{E}$ for training the PLDA model. In general, i-vectors having similar properties are close to each other. One scenario shown in [120] was that most of the PCA projected male i-vectors are close to each others while far from the PCA projected female i-vectors, same for the female i-vectors. Another scenario is visualized in Fig. 5.1 where microphone and telephone recordings are clearly separated. From these examples, we can, therefore, safely assume that the set of i-vectors in $\mathcal{B}$ that has smaller distance from the set of i-vectors in $\mathcal{E}$ can be our desired $\mathcal{S}$.

By selecting i-vectors in $\mathcal{B}$ that are close to the i-vectors in $\mathcal{E}$, we may improve the modeling in the relevant region of i-vector space on the expense of worse modeling in other regions of the i-vectors space. Of-course, the i-vectors of some impostors might be located in the regions where the modeling have been worsen, but our assumption is that these impostors will not be confused with any of the target speakers in $\mathcal{E}$, anyway.

In this thesis, we first use cosine distance to find k number of i-vectors

in $\mathcal{B}$ close to an i-vector in $\mathcal{E}$. This is similar to the neighborhood building part of the classical k-NN approach [37, 22] (See Section 5.2 for details). We show that this method performs remarkably well when the optimal k is known. However, it is difficult to estimate the optimal k. We, therefore, propose a robust way of selecting k based on the *local distance-based outlier factor* (LDOF) [140]. We name our method *flexible* k-NN (fk-NN) [9]. We also explore some variants of k-NN and fk-NN based data selection methods.

Since the training time of the PLDA model is very short (several seconds), fk-NN does not offer a large reduction in computational expense as some data selection methods for UBM training [56] do. However, it improves the verification accuracy. Thanks to the short training time, we can re-train the PLDA model quickly after adding relevant data for newly added enrollment speakers, which would not be practical for UBM training or other offline modeling such as factor loading matrix (e.g., total variability matrix) training.

The organization of the rest of this chapter is as follows: Section 5.2 presents a background study on k-NN. Section 5.3 presents our proposed data selection methods based on k-nearest neighbors and their variants. Section 5.4 and Section 5.5, describe our experimental setup and results, respectively. Section 5.6 includes analysis of different issues regarding experiments.

5.2 Previous Works on k-NN

k nearest neighbors (k-NN) rule is one of the simplest non-parametric machine learning method which is mainly used for classification and regression. No explicit training is necessary in this method. It memorizes all training data points which participate in the classification or discriminant analysis. Therefore, it is considered as a type of memory based learning, instance based learning, or case based learning. It was first investigated by Fix and Hodges [37] for $k \to \infty$ and then by Cover et al. [23] for fixed k.

In k-NN based methods, each training data vector is treated as a reference vector and is labeled either by the class it belongs to or by the discriminant value. The unknown test vector is called a query vector. Given a set Ω with n reference vectors $\{\boldsymbol{x}_i \in \mathbb{R}^D\}_{i=1}^n$, and n labels $\{y_i \in \mathbb{C}$:

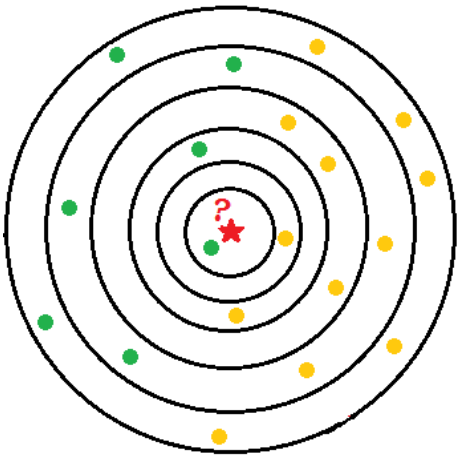

Figure 5.2: Example of k-NN Classification. Red star is the unlabeled query vector, green and yellow circles are reference vectors of Class-1 and Class-2, respectively.

$\{1, 2, ...C\}\}_{i=1}^{n}$ for classification or $y_i \in \mathbb{R}$ for regression, the task of this method is to find a label, y_0, for a query vector $\boldsymbol{x}_0$. For that at first distances from $\boldsymbol{x}_0$ to all $\{\boldsymbol{x}_i\}_{i=1}^{n}$ are estimated, then k closest neighbors of $\boldsymbol{x}_0$ are chosen to build a neighborhood $\mathcal{S}$ for $\boldsymbol{x}_0$.

- **Classification**: In k-NN based classification, y_0 is a class membership, which is predicted by a 'majority rule'. The probability of assigning $\boldsymbol{x}_0$ to class $c \in C$ is:

$$p(c|\boldsymbol{x}_0) = \frac{N_c}{k} \tag{5.1}$$

  where N_c is the number of reference vectors in $\mathcal{S}$ belong to class c. y_0 is set to that c for which $p(c|\boldsymbol{x}_0)$ is the maximum. In the simplest case, $k = 1$, i.e., only the nearest neighbor determines the class of $\boldsymbol{x}_0$ results in $p(c|\boldsymbol{x}_0) = 0$ or 1. During binary classification, generally an odd number is chosen for k in order to avoid ties. For example, as shown in Figure 5.2, the test sample (red star) should be classified either to the Class-1 consisted of green circles or to the Class-2 consisted of yellow circles. If $k = 5$, red star is assigned to Class-2 because there are 3 yellow circles and only 2 green circles inside four inner boundary lines. If $k = 4$, there will be a tie.
- **Regression**: Unlike k-NN based classification, in k-NN based regression y_0 is a real value. The prediction can be done in different ways

such as simple interpolation, local linear regression, local weighted regression, or just averaging of k nearest neighbors property value as follows:

$$y_0 = \sum_{i=1}^{k} w_i y_i \tag{5.2}$$

where w_i could be:

- $w_i = \frac{1}{k}$
- $w_i \sim 1 - ||\boldsymbol{x}_i - \boldsymbol{x}_0||$
- $w_i \sim k - rank||\boldsymbol{x}_i - \boldsymbol{x}_0||$

Since 1951, many investigations have been done for k-NN based classification and regression, some of them are pointed out below:

1. The statistical properties of k-NN classifier like the convergence of the conditional error rate, the limit of the error rate in the case of dependent data, mathematical consistency, etc. were discussed in [23, 22, 40, 30, 51, 50, 29, 21, 89, 121, 66].
2. As k-NN requires to store the whole training set and to estimate distance from each reference vector to the query vector, it may become too costly and very slow when the training set is very large and high-dimensional. Research have been done to mitigate this problem by getting rid of the redundant data or the data which does not provide extra information, or by finding some representatives of the whole training set [55, 44, 2, 87, 134].
3. Similarity measure between the query vector and the reference vectors is an important task in k-NN, since it helps find out informative neighbors which can predict the right label for the query vector. The success of k-NN approach quite depends on the distance function used for measuring similarity. Finding out a better distant metric rather than the common Euclidean, Mahalanobis or Manhattan metric was targeted in [58, 53, 133, 31, 122].
4. The choice of k plays an important role on the performance of the k-NN based approaches, since it decides the volume of the neighborhood, $\mathcal{S}$, and the smoothness of the posterior estimates. If k is too large, it will lead to over-smoothed boundaries, whereas if it is too small, it will lead to noisy decision boundaries. Usually, k is chosen empirically by cross-validation. Depending on each problem, different

values of k are tried, and k with the best performance is chosen. It is a computationally expensive and slow process. Optimization of k rather than cross-validation was investigated in [132, 46, 52].

5. In a high dimensional space, distances between the nearest and the farthest reference vectors from query vectors become almost equal, which causes wrong classification. Wrong classification for the presence of many irrelevant attributes in a vector is often termed as the curse of dimensionality. Mitigating the curse of dimensionality issue was investigated in [69, 139, 117].

Some popular extensions of k-NN approach are described briefly in the following Sections.

5.2.1 Condensed Nearest Neighbor (CNN)

It was proposed by Hart [55] to reduce the number reference vectors from $\Omega : \{\{\boldsymbol{x}_i, y_i\}_{i=1}^{n} \in \{\mathbb{R}^D, \mathbb{C}\}\}$. In this algorithm, a new set of reference vectors, $\Phi \subset \Omega$, is made which is initially empty and gradually filled up by the following steps:

1. Copy the first reference vector from Ω to Φ.
2. Use Φ as the set of reference vectors and Ω as the set of query vectors and do classification by setting $k = 1$ and following the two steps below:
   (a) terminate classification if each query vector in Ω is classified correctly.
   (b) go to Step-3 if any query vector in Ω is classified incorrectly.
3. Copy the incorrectly classified vector in Ω to Φ and go to Step-2.

During classification of a query vector, $\boldsymbol{x}_0$, Φ is used instead of Ω as the set of reference vectors and k can be any value.

5.2.2 Reduced Nearest Neighbor (RNN)

It was an extended version of CNN. It was proposed by Gates [44]. The target of RNN is to make a set of reference vectors, $\Psi \subset \Phi$, by eliminating the reference vectors from Φ which are not affecting the classification result of Ω. The steps of RNN are given below:

1. Copy Φ into Ψ.
2. Remove the first vector from Ψ.
3. Use Ψ to classify all the vectors in Ω using k-NN rule with $k = 1$:
   (a) if all vectors in Ω are classified correctly, go to Step-4.
   (b) if any vector is classified incorrectly, return the vector to Ψ that was removed from Ψ and go to Step-4.
4. If every vector in Ψ is removed once (and possibly replaced) then halt. Otherwise, remove the next vector of Ψ and go to Step-3.

During classification of a query vector, $\boldsymbol{x}_0$, Ψ is used instead of Ω as the set of reference vectors and k can be any value.

5.2.3 Discriminant Adaptive NN (DANN)

It was proposed by Hastie and Tibshirani [58]. In this approach, a distant metric, $\mathbf{\Sigma}$, behaved like a LDA metric is learned, which can be written as follows:

$$\mathbf{\Sigma} = \mathbf{W}^{-1/2}[\mathbf{W}^{-1/2}\mathbf{B}\mathbf{W}^{-1/2} + \epsilon\mathbf{I}]\mathbf{W}^{-1/2} \tag{5.3}$$

where ϵ is some small tuning parameter to be determined, $\mathbf{I}$ is a $D \times D$ dimensional identity-matrix, $\mathbf{W}$ and $\mathbf{B}$ are the $D \times D$ dimensional *within* and *between* sum-of-squares matrices, respectively. These sum-of-squares matrices are estimated using the neighborhood $\mathcal{S}$ created by K_L nearest neighbors of the query vector $\boldsymbol{x}_0$. Initially distant metric, $\mathbf{\Sigma}$, is set to $\mathbf{I}$. Then it is updated iteratively by using updated $\mathbf{B}$ and $\mathbf{W}$ which are estimated as follows:

$$\mathbf{B} = \sum_{j=1}^{J} \alpha_j (\bar{\boldsymbol{x}}_j - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_j - \bar{\boldsymbol{x}})' \tag{5.4}$$

$$\mathbf{W} = \frac{\sum_{j=1}^{J} \sum_{i \in J} \alpha_j (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_j)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_j)'}{\sum_{i=1}^{K_L} w_i} \tag{5.5}$$

where $J \subset C$ contains the classes inside $\mathcal{S}$; $\bar{\boldsymbol{x}}_j$ is the weighted mean of the N_j observations in the j-th class and

$$\alpha_j = \frac{\sum_{i \in j} w_i}{\sum_{i=1}^{K_L} w_i} \tag{5.6}$$

$$w_i = [1 - (d_i/h)^3]^3 I(|d_i| < h) \tag{5.7}$$

where,

$$\begin{aligned} d_i &= d(\boldsymbol{x}_0, \boldsymbol{x}_i) \\ &= ||\boldsymbol{\Sigma}^{1/2}(\boldsymbol{x}_i - \boldsymbol{x}_0)|| \end{aligned} \tag{5.8}$$

$$h = \max_{\boldsymbol{x}_i \in \mathcal{S}} d(\boldsymbol{x}_0, \boldsymbol{x}_i) \tag{5.9}$$

After learning $\boldsymbol{\Sigma}$ by an iterative process, K_G nearest neighbors of $\boldsymbol{x}_0$ are found by estimating distance using Eq. 5.8. $\boldsymbol{\Sigma}$ is different for different $\boldsymbol{x}_0$. Note that K_L and K_G are two different values. K_L is used for estimating the distant metric, whereas K_G is used for classifying $\boldsymbol{x}_0$. K_L should be reasonably large, since neighborhood $\mathcal{S}$ created by K_L is used to estimate covariance matrices, whereas K_G is set to a small value in order to avoid bias. Another point is that except two tuning parameters, K_L and K_G, there is another parameter that needs to be tuned and that is ϵ.

5.2.4 Weight Adjusted k-NN (WAKNN)

It was proposed by Han et al. [53] for categorizing documents. In this method at first the mutual information of each word in the training document set is estimated as follows:

$$MI(w) = \sum_{c \in C}(p(c, w) \log \frac{p(c, w)}{p(c)p(w)} + p(c, \bar{w}) \log \frac{p(c, \bar{w})}{p(c)p(\bar{w})}) \tag{5.10}$$

where $p(c)$ is the probability of class c, $p(w)$ is the probability of the presence of word w, and $p(\bar{w})$ is the probability of the absence of word w, and $p(c, w)$, and $p(c, \bar{w})$ are joint probabilities. Then the importance of each word is learned using mutual information as the initial value of weight for that word and applying weight adjustment steps which is a iterative procedure. During classification, k-NN of the query document from the training documents are found using the following weighted cosine similarity measure:

$$cos(X, Y, \alpha) = \frac{\sum_{w \in W}(X_w \times \alpha_w) \times (Y_w \times \alpha_w)}{\sqrt{\sum_{w \in W}(X_w \times \alpha_w)^2} \times \sqrt{\sum_{w \in W}(Y_w \times \alpha_w)^2}} \tag{5.11}$$

where X_w and Y_w are normalized within-document word frequency of word w for a reference document X and a query document Y, respectively, and α_w is the weight of word w.

5.2.5 Adaptive Metric NN (ADAMENN)

It was proposed by Domeniconi et al. [31] in order to minimize bias introduced by high dimensional finite number of reference vectors. A new distant metric based on Chi-square distance analysis was introduced for producing a neighborhood which is elongated along less relevant feature dimensions and constricted along most influential ones. According to the new distant metric, the distance between a query vector $\boldsymbol{x}_0$ and a reference vector $\boldsymbol{x}_i$ can be written as:

$$d(\boldsymbol{x}_0, \boldsymbol{x}_i) = \sqrt{\sum_{d=1}^{D} w_d (\boldsymbol{x}_0^d - \boldsymbol{x}_i^d)^2} \tag{5.12}$$

where $\{\boldsymbol{x}_i \in \mathbb{R}^D\}_{i=1}^n$ with class labels $\{y_i \in \mathbb{C}\}_{i=1}^n$ and w_d is the weight associated with feature d of $\boldsymbol{x}_0$, defined as:

$$w_d = \exp\left(\beta \frac{1}{r_d}\right) / \sum_{d=1}^{D} \exp\left(\beta \frac{1}{r_d}\right) \tag{5.13}$$

where β is a parameter to control the influence of r_d on w_d, and r_d is a measure of feature relevance of $\boldsymbol{x}_0$ with respect to its neighborhood $\mathcal{S}_0$ created by K_0 nearest neighbors and r_d is defined as:

$$r_d = \frac{1}{K_0} \sum_{\boldsymbol{z} \in \mathcal{S}_0} \sum_{c=1}^{C} \frac{[\mathcal{P}(c|\boldsymbol{z}) - \hat{\mathcal{P}}(c|\boldsymbol{x}_0^d = \boldsymbol{z}_d)]^2}{\hat{\mathcal{P}}(c|\boldsymbol{x}_0^d = \boldsymbol{z}_d)} \tag{5.14}$$

where $\mathcal{P}(c|\boldsymbol{z})$ and $\bar{\mathcal{P}}(c|\boldsymbol{x}_0^d = \boldsymbol{z}_d)$ are estimated using $\Omega : \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ and two neighborhoods $\mathcal{S}_1$ and $\mathcal{S}_2$ centered at $\boldsymbol{z} \in \mathcal{S}_0$. The size of $\mathcal{S}_1$ and $\mathcal{S}_2$ are K_1 and K_2, respectively. During classification k different from K_0, K_1, and K_2 is used to create a neighborhood $\mathcal{S}$. This approach has six adjustable tuning parameters: K_0, K_1, K_2, k, β and L: the number of points within the Δ intervals used for estimating $\bar{\mathcal{P}}(c|\boldsymbol{x}_0^d = \boldsymbol{z}_d)$.

5.2.6 Large Margin NN (LMNN)

This was proposed by Weinberger et al. [133]. In this approach, a Mahanalobis distance metric, **M**, is learned by using $\Omega : \{\{\boldsymbol{x}_i, y_i\}_{i=1}^n \in \{\mathbb{R}^D, C\}\}$ and the following semidefinite programming:

$$\textbf{Minimize } (1-\alpha) \sum_{i, j \rightsquigarrow i} d(\boldsymbol{x}_i, \boldsymbol{x}_j) + \alpha \sum_{i, j \rightsquigarrow i, l} (1 - y_{il}) \xi_{ijl} \tag{5.15}$$

subject to:

1. $d(\boldsymbol{x}_i, \boldsymbol{x}_l) - d(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 1 - \xi_{ijl}$
2. $\xi_{ijl} \geq 0$ and
3. $\mathbf{M} \succeq 0$.

where $\alpha \in [0, 1]$ is a balancing factor which can be tuned by cross validation, ξ_{ijl} is the slack variable, $\boldsymbol{x}_j$ and $\boldsymbol{x}_l$ are the target neighbor and impostor of any data point $\boldsymbol{x}_i$, respectively. Both $\boldsymbol{x}_j$ and $\boldsymbol{x}_l$ are included in the set of k nearest neighbors of $\boldsymbol{x}_i$, however, $y_j = y_i$, whereas $y_l \neq y_i$. The last constraint ensures that $\mathbf{M}$ is positive semi-definite. The distance between any two points is estimated as:

$$d(\boldsymbol{a}, \boldsymbol{b}) = (\boldsymbol{a} - \boldsymbol{b})'\mathbf{M}(\boldsymbol{a} - \boldsymbol{b}) \tag{5.16}$$

During classifying any query vector, $\boldsymbol{x}_0$, k-NN of $\boldsymbol{x}_0$ is found out by using distance measure in Eq. 5.16 using learned $\mathbf{M}$.This approach is called *large margin nearest neighbor* since it seeks a large margin that separates examples from different classes, while keeping a close distance between nearest neighbors that have the same class labels. The novelty of this method is that it does not try to minimize the distance between all examples that share the same labels, but only to those that are specified as target neighbors.

5.2.7 Informative KNN (IKNN)

Song et al. [122] introduced a new distant metric using the measure of importance of reference vectors to a query vector. They named the measure of importance as *informativeness*, $\mathcal{I}$. Based on the new distance metric, they proposed locally informative KNN (LI-KNN) and globally informative KNN (GI-KNN). In LI-KNN, at first k nearest reference vectors to a query vector is selected using Euclidean distance. Among k nearest reference vectors, I number of most informative reference vectors are selected for deciding class label of the query vector. Let the neighborhood of a query vector, $\boldsymbol{x}_0$, created by k nearest neighbors $\{\boldsymbol{x}_i \in \mathbb{R}^D, y_i \in \mathbb{C}\}_{i=1}^k$ with $\{\mathbb{C} : 1, 2, 3...., C\}$ be $\mathcal{S}$. A neighbor, $\boldsymbol{x}_i \in \mathcal{S}$, is treated to be informative for the query vector, $\boldsymbol{x}_0$, if $\boldsymbol{x}_i$ is close to $\boldsymbol{x}_0$, and far away from the reference vectors with different class labels. The *informativeness* of $\boldsymbol{x}_i$ for $\boldsymbol{x}_0$ can be defined as:

$$\mathcal{I}_{\boldsymbol{x}_0}(\boldsymbol{x}_i) = -\log(1 - \mathcal{P}(\boldsymbol{x}_i|\boldsymbol{x}_0)) * \mathcal{P}(\boldsymbol{x}_i|\boldsymbol{x}_0),\ i = 1, 2, 3, ...k \tag{5.17}$$

where $\mathcal{P}(\boldsymbol{x}_i|\boldsymbol{x}_0)$ is the probability that $\boldsymbol{x}_i$ is informative for $\boldsymbol{x}_0$ and defined as:

$$\mathcal{P}(\boldsymbol{x}_i|\boldsymbol{x}_0) = \frac{1}{Z_i}\left\{p(\boldsymbol{x}_i|\boldsymbol{x}_0)^{\eta}\left(\prod_{i=1}^{k}(1-p(\boldsymbol{x}_i|\boldsymbol{x}_0)\mathbb{I}_{y_i\neq y_k})\right)^{(1-\eta)}\right\} \tag{5.18}$$

where the indicator $\mathbb{I}[.]$ equals to 1 if the condition is met and 0 otherwise, η is a balancing factor and Z_i is a normalization factor defined as:

$$Z_i = \sum_{i=1}^{k} p(\boldsymbol{x}_i|\boldsymbol{x}_0). \tag{5.19}$$

where $p(\boldsymbol{x}_i|\boldsymbol{x}_0)$ is defined as:

$$p(\boldsymbol{x}_i|\boldsymbol{x}_0) = \exp(-\frac{\sum_{d=1}^{D} w_d(\boldsymbol{x}_{0d}-\boldsymbol{x}_{id})^2}{\gamma}), \tag{5.20}$$

where $\gamma > 0$ and w_d is obtained by averaging over all classes' weights w_{dc}, which is calculated using the variance of all points in each class c at feature d, denoted by Var$(\boldsymbol{x}_{dc})$.

$$w_d = \frac{1}{C}\sum_{c=1}^{C} w_{dc} = \frac{1}{C}\sum_{c=1}^{C} \text{Var}(\boldsymbol{x}_{dc}) \text{ where } d = 1, 2, 3, ...D, \tag{5.21}$$

In GI-KNN, more emphasis is put on those reference vectors that are globally informative. For that an optimum weight vector $\boldsymbol{A}$ is estimated using all reference vectors and query vectors by an iterative process and that weight vector is used to measure distance between the query vector and a reference vector.

Shortcomings of k-NN variants

Despite the success for improving classification performance, the above mentioned extended k-NN have several shortcomings. The weakness of CNN and RNN is that they do not ensure that the resulting condensed neighborhood is the smallest consistent set. Depending on the order in which the reference vectors are selected, different condensed sets with different sizes will be obtained. All sets will be consistent but many reference vectors will not keep consistency property in the condensed set. DANN introduces two new parameters, K_L and ϵ; ADAMENN has five new parameters to be tuned that could be cause of overfitting, LMNN applies semidefinite programming for

the optimization problem, WAKNN is designed specifically for text categorization, IKNN has two extra parameters to be tuned and measuring informativeness is a time consuming procedure and so on. Additionally, choosing the proper value of k is a crucial task in all approaches.

5.2.8 Optimization of k

Even though the success of a k-NN based approach to some extent depends on the choice of k, it is very hard to find any works in literature regarding this issue. Here we summarize some of the works found in literature:

1. Fukunaga and Hostetler [39] developed a functional form for the optimum k in terms of the sample size, the dimensionality of the observation space, and the underlying probability distribution [39].
2. Enas et al. [32] showed an efficient adaptive rule which selects k by iteratively maximizing the local Mahalanobis distance .
3. Wang et al. [132] argued that each neighborhood contains some degree of support for all classes. If all these supports are aggregated, we could end up with a less biased classification in the sense that it is not too dependent on a single value for k.
4. In [52], Hall et al. used Poisson and Binomial distributions for choosing value of k.
5. Ghosh mentioned that popular cross-validation techniques often fail in selecting k due to the presence of multiple minimizers of the estimated misclassification rate. He proposed a Bayesian method in [46]. According to his method, the optimum k is k_0 so that

$$\alpha(k_0) > \alpha(k),\ \forall k \neq k_0, \tag{5.22}$$

where $\alpha(k)$ is the accuracy index defined as:

$$\alpha(k) = \sum_{c=1}^{C} \pi_c \int S(c|k) f_c(\boldsymbol{x}_0) d\boldsymbol{x}_\mathbf{0}, \tag{5.23}$$

where f_c, π_c and $S(c|k)$ are the density function, prior probability, and strength function of class c, respectively, which can be defined as:

$$\pi_c = \frac{n_c}{n} \tag{5.24}$$

$$S(c|k) = \int_{p_c=max\{p_1,p_2,...,p_C\}} \zeta(\boldsymbol{p}|k,\boldsymbol{t}_k)d\boldsymbol{p} \tag{5.25}$$

where $p_c = p(c|\boldsymbol{x}_0)$, $\boldsymbol{p} = (p_1, p_2, ..., p_C)$, $\sum_{c=1}^{C} p_c = 1$, $\boldsymbol{t}_k = (t_{1_k}, t_{2_k}, ..., t_{C_k})$, $\sum_{c=1}^{C} t_{c_k} = k$, t_{c_k} is the number of neighbors among k neighbors come from the c-th class, and $\zeta(\boldsymbol{p}|k, t_k)$ is the conditional probability of $\boldsymbol{p}$ for some fixed k and $\boldsymbol{t}_k$ defined as:

$$\zeta(\boldsymbol{p}|k,\boldsymbol{t}_k) = \frac{\xi_k(\boldsymbol{p})\phi(\boldsymbol{t}_k|\boldsymbol{p},k)}{\int \xi_k(\boldsymbol{p})\psi(\boldsymbol{t}_k|\boldsymbol{p},k)d\boldsymbol{p}} \tag{5.26}$$

where $\xi_k(\boldsymbol{p})$ is the prior distribution of $\boldsymbol{P}$ in the neighborhood around $\boldsymbol{x}_0$ and $\psi(\boldsymbol{t}_k|\boldsymbol{p},k)$ is the conditional probability of $\boldsymbol{t}_k$ for given $\boldsymbol{p}$ and k, defined as:

$$\psi(\boldsymbol{t}_k|\boldsymbol{p},k) = \binom{k}{t_{1_k}, t_{2_k}, ..., t_{C_k}} \prod_{c=1}^{C} p_c^{t_{c_k}} \tag{5.27}$$

Even though above mentioned studies have been done, their findings did not get popularity. Most of the time, k is decided empirically. That means different values of k are tried, and k with the best performance is chosen. Most of the time, a cross-validation approach is applied.

5.3 i-Vector Selection

Unlike classification or regression discussed in Section 5.2, in this thesis we use k-NN to find out a set of reference vectors, $\boldsymbol{X}$, which follows the same distribution, as the set of query vectors, $\boldsymbol{X}_0$, so that $\boldsymbol{X}$ can be used to provide prior to reduce nuisance attributes from $\boldsymbol{x}_0 \in \boldsymbol{X}_0$. In another way, we can say that we use k-nearest neighbors only, not the rule based on k-nearest neighbors. In our case, $\{\boldsymbol{x}_i \in \boldsymbol{X}\}_{i=1}^{n}$ are unlabeled data, and both $\boldsymbol{X}$ and $\boldsymbol{X}_0$ are i-vectors extracted from speech segments. $\boldsymbol{X}$ are prepared from the background dataset, $\mathcal{B}$ and $\boldsymbol{X}_0$ are from the enrollment set, $\mathcal{E}$.

Section 5.3.1 describes our data selection approach based on k nearest neighbors along with its shortcomings. Our approach in order to overcome those shortcoming is described in Section 5.3.2. Few variants of our proposed approaches are presented in Section 5.3.3. Different kinds of issues regarding our proposed data selection approaches are discussed in Section 5.3.4.

5.3.1 k-NN for i-vector Selection

Let the sets of i-vectors, $\boldsymbol{\omega}$, for the PLDA modeling, for enrollment speakers, and for authentication be $\mathcal{B}$, $\mathcal{E}$, and $\mathcal{A}$, respectively. Our target is to select a subset, $\mathcal{S} \subset \mathcal{B}$, that is more suitable for $\mathcal{E}$ than the whole set $\mathcal{B}$.

Algorithm of k-NN based i-vector Selection

Let $\mathcal{S}_e^k \subset \mathcal{B}$ be the set of the k-nearest neighbors of an enrollment i-vector, $\boldsymbol{\omega}_e$. Our target is to find $\mathcal{S}_e^k \subset \mathcal{B}$ for all $\boldsymbol{\omega}_e \subset \mathcal{E}$ in order to build $\mathcal{S} \subset \mathcal{B}$. The steps of our data selection process using *k*-NN are given as:

1. Set the value of k.
2. For each $\boldsymbol{\omega}_e \in \mathcal{E}$, find $\mathcal{S}_e^k$.
   (a) Estimate the distance from ω_e to each $\boldsymbol{\omega}_b \in \mathcal{B}$, i.e., $\mathrm{dist}(\boldsymbol{\omega}_e, \boldsymbol{\omega}_b)$.
   (b) Sort $\mathrm{dist}(\boldsymbol{\omega}_e, \boldsymbol{\omega}_b)$ in ascending order.
   (c) Put the k-nearest neighbors of $\boldsymbol{\omega}_e$ from the set of $\boldsymbol{\omega}_b \in \mathcal{B}$ into $\mathcal{S}_e^k$.
3. Take the unique set of i-vectors from $\{\mathcal{S}_e^k\}_{\forall e}$ to get $\mathcal{S}$.

Problems in k-NN based i-vector Selection

We can choose the value of k from a range of values for which we can get the best verification accuracy on a development set. However, a careful tuning of k in this way might be computationally expensive. The second problem is that k may vary from database to database. Therefore, one k does not guarantee good result in all evaluation sets. Furthermore, the size and the spreadness of $\mathcal{E}$ compared to the spreadness of $\mathcal{B}$ may affect the number of selected i-vectors. If the i-vectors in $\mathcal{E}$ are close to each other compared to the typical distance between the i-vectors in $\mathcal{B}$, then every $\boldsymbol{\omega}_e \in \mathcal{E}$ will select almost the same $\boldsymbol{\omega}_b \in \mathcal{B}$. In such case, if the size of $\mathcal{E}$ is very small, we need a large k in order to get a sufficient amount of i-vectors for training a good PLDA model. On the other hand, if the size of $\mathcal{E}$ is large, then a large k may select unnecessary data. Therefore, if we use a k optimised for a different $\mathcal{E}$, we may not get a sufficient amount of i-vectors for training a good PLDA model, or we may end up covering almost the whole training set, $\mathcal{B}$. Another more complicated problem is that the i-vectors in $\mathcal{S}_e^k$ might be much closer to each other than they are to $\boldsymbol{\omega}_e \in \mathcal{E}$. In this case, the i-vectors in $\mathcal{S}_e^k$ form

a cluster, and $\boldsymbol{\omega}_e$ becomes its *outlier*. In such a case, $\mathcal{S}_e^k$ cannot be expected to improve modeling of the region surrounding $\boldsymbol{\omega}_e$.

In order to solve these problems, we propose a modification of the k-NN method, which we name *flexible* k-NN (fk-NN) [9]. In this method, we first use the LDOF defined in the next section to measure to what extent $\boldsymbol{\omega}_e$ deviates from the cluster made by $\mathcal{S}_e^k$. We then increase k until all $\boldsymbol{\omega}_e \in \mathcal{E}$ lie inside the cloud of nearest neighbors according to the LDOF criteria. Our proposed fk-NN helps to decide k based on the nature of the target $\mathcal{E}$, not on any development set.

5.3.2 Flexible k-NN (fk-NN)

LDOF

In data mining applications, LDOF proposed by [140] is used for capturing the outlierness of an object among a scattered neighborhood. In this thesis, we use it to control the value of k in the k-NN based data selection process. In order to estimate to what degree $\boldsymbol{\omega}_e$ is an outlier with respect to its k nearest neighbors, LDOF checks the average distance between $\boldsymbol{\omega}_e$ and the k nearest neighbors, as well as the average among all the neighbor pairs. If the former is large compared to the latter, $\boldsymbol{\omega}_e$ can be regarded as an outlier. Formally, LDOF of $\boldsymbol{\omega}_e$ given k is defined as

$$\mathrm{LDOF}_e^k = \frac{d_e^k}{D_e^k}, \tag{5.28}$$

where d_e^k is the k-NN distance of $\boldsymbol{\omega}_e$ and D_e^k is the k-NN inner distance of $\boldsymbol{\omega}_e$'s neighbourhood, which are defined as:

$$d_e^k = \frac{1}{k} \sum_{\boldsymbol{\omega}_i \in \mathcal{S}_e^k} \mathrm{dist}(\boldsymbol{\omega}_e, \boldsymbol{\omega}_i), \tag{5.29}$$

$$D_e^k = \frac{1}{k(k-1)} \sum_{\boldsymbol{\omega}_i, \boldsymbol{\omega}_j \in \mathcal{S}_e^k, i \neq j} \mathrm{dist}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j). \tag{5.30}$$

As shown in Fig. 5.3, LDOF captures the degree to which $\boldsymbol{\omega}_e$ deviates from its neighborhood $\mathcal{S}_e^k$. When $\mathrm{LDOF}_e^k \leq 1$, we can say that $\boldsymbol{\omega}_e$ is surrounded by the cloud created by the i-vectors of $\mathcal{S}_e^k$. Notice that if $k = 1$, D_e^k is undefined, therefore, LDOF cannot be calculated.

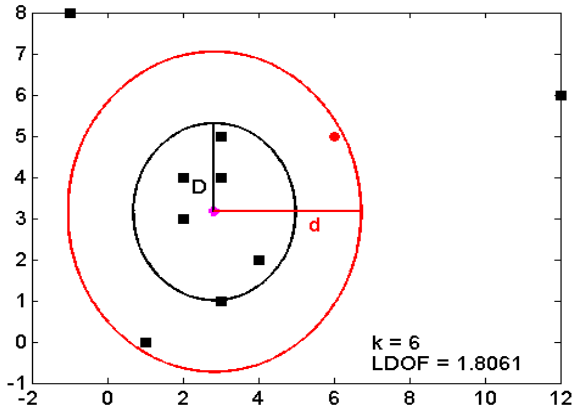

Figure 5.3: The outlierness of a synthetic two-dimensional i-vector, $\boldsymbol{\omega}_e \in \mathcal{E}$, with respect to its six neighbours, $\boldsymbol{\omega}_b \in \mathcal{B}$, according to the LDOF criteria. Here red dot: $\boldsymbol{\omega}_e \in \mathcal{E}$, black square: $\boldsymbol{\omega}_b \in \mathcal{B}$, magenta dot: center of six $\boldsymbol{\omega}_b \in \mathcal{B}$. Among nine ω_b, three are on the boundary lines.

Algorithm of fk-NN

As argued in above, in order for each enrollment i-vector to be well represented by the training data, we would like to make sure that each enrollment i-vector is an inlier its neighborhood. The most naive way to ensure this is to select an individual value of k, k_e, for each $\boldsymbol{\omega}_e$, as the minimum from among those k's which suffice $\mathrm{LDOF}_e^k \leq 1$. However, LDOF is not robustly estimated for values of k much smaller than the dimension of the data. In order to use LDOF in a cautious way, we therefore first find the minimum k which suffice $\mathrm{LDOF}_e^k \leq 1$ for all $\boldsymbol{\omega}_e$. We then use this k for all $\boldsymbol{\omega}_e$. We call this method fk-NN [9]. The fk-NN algorithm is as follows:

1. Set the LDOF threshold, θ, so that $0 < \theta \leq 1$.
2. Set $k = 2$.
3. For each $\omega_e \in \mathcal{E}$,
   (a) Find $\mathcal{S}_e^k$.
   (b) Estimate LDOF_e^k.
4. If any $\mathrm{LDOF}_e^k \geq \theta$, then
   (a) $k = k + 1$.
   (b) Go to Step-3

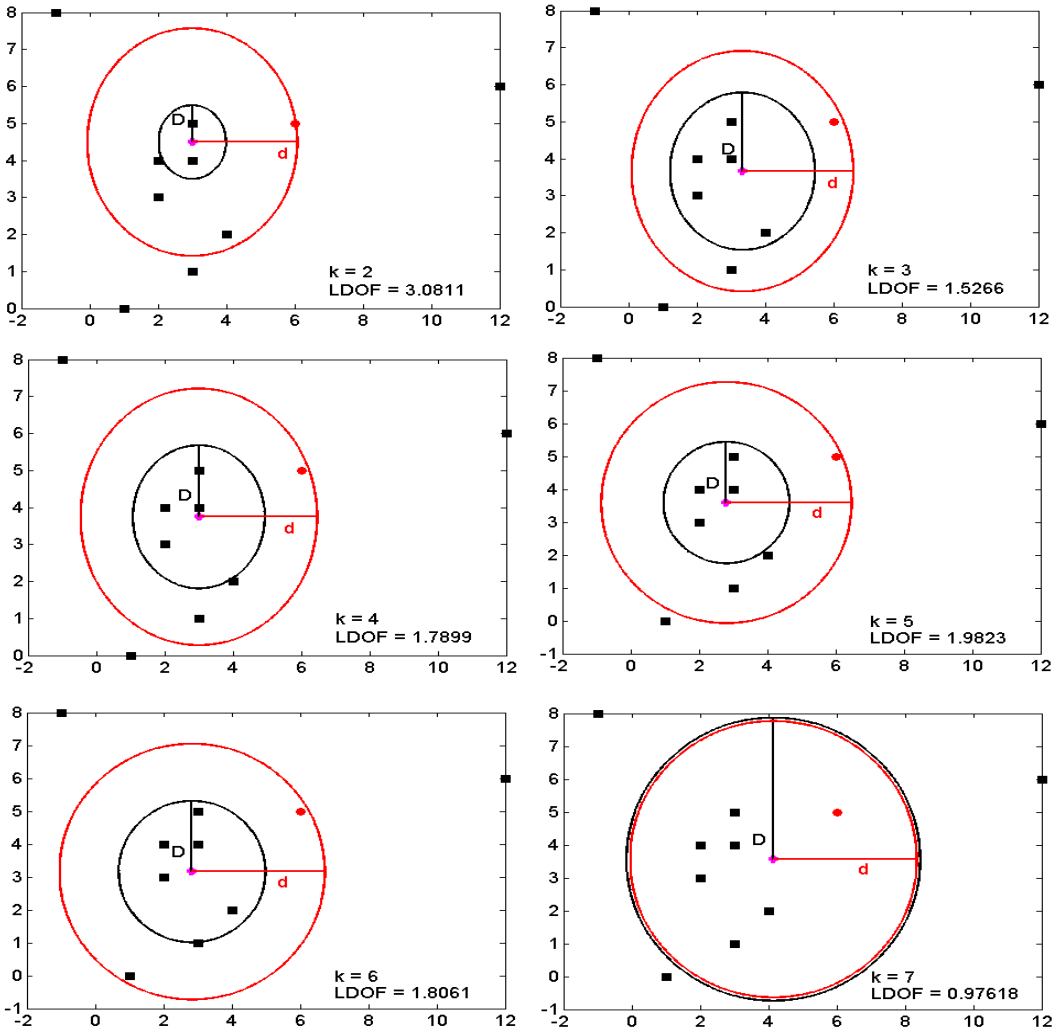

Figure 5.4: Estimation of k for a synthetic two-dimensional i-vector, $\boldsymbol{\omega}_e \in \mathcal{E}$, by using LDOF value. Here red dot: $\boldsymbol{\omega}_e \in \mathcal{E}$, black square: $\boldsymbol{\omega}_b \in \mathcal{B}$, magenta dot: center of i-vectors in $\mathcal{S}_e^k$. For $k = 7$ the $\text{LDOF}_e^k < 1$.

5. Take the unique set of i-vectors from $\{\mathcal{S}_e^k\}_{\forall e}$ to get $\mathcal{S}$.

In order to avoid an extra parameter to tune, we set $\theta = 1$ in our experiments. Fig. 5.4 shows how the LDOF value is used to decide the value of k in fk-NN for a synthetic two-dimensional enrollment i-vector, $\boldsymbol{\omega}_e$. In this figure, k is gradually increases until $\text{LDOF}_e^k \geq \theta = 1$ which happens for $k = 7$. The highest value of k obtained in this way for any enrollment i-vector will be the value of k according to fk-NN.

5.3.3 k-NN and fk-NN Variants

Individual k-NN (ik-NN)

As mentioned in Section 5.3.2, it is difficult to estimate an individual k for each $\boldsymbol{\omega}_e$ using LDOF since LDOF for small values of k is not robostly estimated. Here, we propose a variant of fk-NN using the difference between LDOF_e^k's. Let ΔLDOF_e^k be the absolute difference between LDOF_e^k and LDOF_e^{k-1}, and γ be the threshold for ΔLDOF_e^k. Then, for each $\boldsymbol{\omega}_e$, we increase k_e as long as $\text{LDOF}_e^k \geq 1$ and $\Delta\text{LDOF}_e^k \leq \gamma$. Here, we use absolute difference since we assume that the differences converge to 0 as k increases without necessarily being negative for all k. We refer to this method as *individual* k-NN (ik-NN). The steps of ik-NN are given below:

1. For each $\omega_e \in \mathcal{E}$,
   (a) Set $k_e = 2$.
   (b) Find $\mathcal{S}_e^k$.
   (c) Estimate LDOF_e^k.
   (d) If $\text{LDOF}_e^k \geq 1$ and $\Delta\text{LDOF}_e^k \leq \gamma$, then
      i. $k_e = k_e + 1$.
      ii. Go to Step-1b
2. Take the unique set of i-vectors from $\{\mathcal{S}_e^k\}_{\forall e}$ to get $\mathcal{S}$.

Averaged Enrollment i-Vectors

Up until now we have discussed the scenario where each target speaker has one enrollment i-vector. However, in some cases such as in the NIST SRE12 data set, the target speakers sometimes have several enrollment i-vectors. For such cases, we propose an alternative strategy where at first we average the enrollment i-vectors of each speaker. Then we use the averaged i-vectors for data selection with k-NN or fk-NN in the normal way. We denote the methods as a-k-NN and a-fk-NN, respectively.

Adding All Sessions from Selected Speakers

Having many sessions per speaker is important for reliable estimation of both the speaker variability, $\mathbf{V}$, and the channel variability, $\boldsymbol{\Sigma}$. (Consider for example the extreme case of having only one session per traning speaker in

which speaker and channel variability cannot be separated in PLDA training.) Our data selection approaches (k-NN, fk-NN, ik-NN) are, however, unlikely to select all sessions of each selected speaker. In order to avoid this problem, we propose a variant of our data selection methods where we first apply k-NN, fk-NN or ik-NN, and then add all discarded i-vectors from the speakers in $\mathcal{S}$. We call this method k-NN-s, fk-NN-s or ik-NN-s. Since adding more i-vectors may lose the theoretical justification for fk-NN-s and ik-NN-s, we focus on k-NN-s in this thesis.

5.3.4 Issues Related to Data Selection

Distance metric

The choice of the distance measure is an important issue in both k-NN and fk-NN. Various measures can be used to compute the distance between two i-vectors. From Fig. 5.1, we can say that the Euclidean distance could be a good choice. However, since we are using length-normalized i-vectors for PLDA modeling, it would be inconsistent to use the Euclidean distance without length normalization in the i-vector selection phase. Because, some $\boldsymbol{\omega}_b \in \mathcal{B}$ that are close to an $\boldsymbol{\omega}_e \in \mathcal{E}$ before length-normalization may not be close after length-normalization. Thus wrong i-vectors may be selected which will deteriorate the performance of k-NN based system. Our preliminary experiment using the Euclidean distance supported this fact. In Fig. 5.5, the same i-vectors shown in Fig. 5.1, are shown after length normalization. As can be seen, there is an obvious directional separation between the i-vectors extracted from phone- and microphone-utterances. Therefore, the cosine distance,

$$\text{dist}_{\text{cos}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j) = 1 - \frac{\boldsymbol{\omega}_i^T \boldsymbol{\omega}_j}{\sqrt{\boldsymbol{\omega}_i^T \boldsymbol{\omega}_i \boldsymbol{\omega}_j^T \boldsymbol{\omega}_j}} \tag{5.31}$$

could be a good choice. When the i-vectors are length-normalized, i.e., $\boldsymbol{\omega}_i^T \boldsymbol{\omega}_i = 1$, the relation between the two distance metrics is given by,

$$\begin{aligned} \text{dist}_{\text{euc}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j) &= \sqrt{(\boldsymbol{\omega}_i - \boldsymbol{\omega}_j)^T (\boldsymbol{\omega}_i - \boldsymbol{\omega}_j)} \\ &= \sqrt{2 \times \text{dist}_{\text{cos}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j)}, \end{aligned} \tag{5.32}$$

where $\text{dist}_{\text{euc}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j)$ is the Euclidean distance and $\text{dist}_{\text{cos}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j)$ is the cosine distance between any two i-vectors, $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$. Since this function is

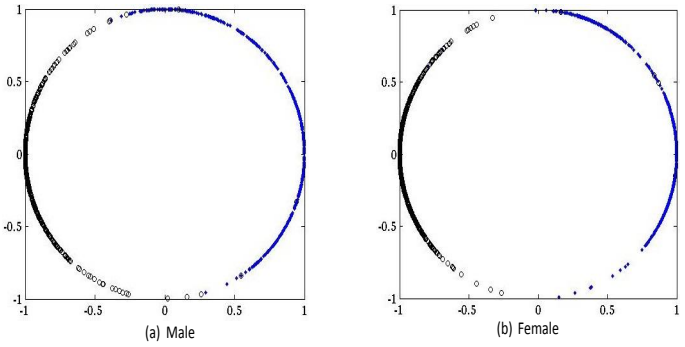

Figure 5.5: Plot of the length-normalized i-vectors after applying a two dimensional PCA-projection. Black circles represent i-vectors extracted from the phone-utterances and blue dots represent i-vectors extracted from the microphone-utterances. (a) There are 1270 male i-vectors among them 648 are from the phone utterances and 622 are from the microphone utterances. (b) There are 1993 female i-vectors among them 1140 are from the phone-utterances and 853 are from the microphone-utterances.

monotonically rising, it does not make any difference which of the two distance metrics we use when i-vectors are length-normalized. In this study, we use the cosine distance for both k-NN and fk-NN. A more detailed analysis of distance metric will be a part of future work.

Domain Adaptation

If we can use $\mathcal{E}$ for selecting relevant data from $\mathcal{B}$, we can also use $\mathcal{E}$ for *domain* adaptation. The most trivial domain adaptation approach is to add i-vectors of $\mathcal{E}$ (i.e., the in-domain data) to the PLDA training data (i.e., the out-domain data), and re-train the model. Domain adaptation adjusts the model to be more similar to $\mathcal{E}$ but equal emphasis on the relevant data $\mathcal{S} \subset \mathcal{B}$ and the irrelevant data $\mathcal{B} \setminus \mathcal{S}$. Data selection improves the modeling in the region close to $\mathcal{E}$ on the expense on regions far from $\mathcal{E}$ but does not use $\mathcal{E}$ for training. Data selection and domain adaptation can therefore be expected to be complementary, unless the enrollment set is so large that it is enough for PLDA training by itself.

Unseen Impostors

A possible concern with the idea of data selection based on $\mathcal{E}$ is that this may reduce the performance for non-target trials involving *unknown* impostors, i.e., impostors who are not in $\mathcal{E}$. The reason for this concern is that by selecting training data close to $\mathcal{E}$, the modeling of impostors in regions far away from $\mathcal{E}$ may deteriorate. However, our assumption is that impostors who are far away from $\mathcal{E}$ will not be confused with speakers in $\mathcal{E}$ anyway. In order to verify that data selection does not reduce the performance for non-target trials involving unknown impostors, we experimentally compare the performance of data selection when all the impostors are unknown and when all the impostors are *known*, i.e., they are one of the speakers in $\mathcal{E}$ in Section 5.6.6.

5.4 Experimental Set-up

In order to examine the effect of $\mathcal{S} \subset \mathcal{B}$ selected by k-NN, fk-NN and their variants on the performance of the PLDA model, we conducted experiments on several recent NIST SREs. In these evaluations, there are conditions focusing on telephone speech, microphone speech or both. We restricted our experiments to conditions having authentication sets, $\mathcal{A}$s, containing telephone data only. We chose to work on the telephone data because the amount of microphone data in our $\mathcal{B}$ was small. During the development of our baseline system, we found that it was beneficial to exclude i-vectors estimated from utterances of $\mathcal{B}$ that were distorted by *echo*, or *crosstalk*, or *background noise* based on meta-data. However, in reality meta-data may not always be available. Therefore, we considered two background datasets, $\mathcal{B}_1$ including distorted speech and $\mathcal{B}_2$ excluding distorted speech for PLDA training. For our convenience, we denote $\mathcal{B}_1$ as $\mathcal{R}$ and $\mathcal{B}_2$ as $\mathcal{C}$ in the later sections.

The details of our development set, $\mathcal{D}$, and evaluation sets, $\mathcal{V}$s, are given in Section 5.4.1. Section 5.4.2 describes background datasets used for training UBMs, i-vector extractor, $\boldsymbol{T}$ and PLDA models. Extracting features, training models and tuning k are discussed in Section 5.4.3 and in Section 5.4.4. Evaluation metrics are described in Section 5.4.5.

5.4.1 Development Set and Evaluation Set ($\mathcal{D}$ and $\mathcal{V}$)

We used the NIST SRE 2006 core task (SRE06) as $\mathcal{D}$, in particular for tuning k in k-NN. The NIST SRE 2008 core task condition-6 (SRE08), the NIST SRE 2010 core task condition-5 (SRE10), and the NIST SRE 2012 core task condition-2, -4 and -5 (SRE12) were used as $\mathcal{V}$s. Among the three $\mathcal{V}$s, SRE12 has noisy $\mathcal{A}$s, whereas the other two are considered to have clean $\mathcal{A}$s.

SRE06, SRE08 and SRE10 have only clean speech files in $\mathcal{A}$. In $\mathcal{E}$, there are only one speech file for training a model for each target speaker. Each speech file of $\mathcal{E}$ is from approximately five minutes long conversational telephone speech. Some speakers in $\mathcal{E}$ have multiple model IDs. Therefore, the number of speaker models, #M, is larger than the number of speakers, #Es. SRE12 has noisy $\mathcal{A}$s. In SRE12(c5), all speech files of $\mathcal{A}$ have intentionally been collected in a noisy environment. In SRE12(c4), the files of $\mathcal{A}$ have added noise. SRE12(c2) includes all the trials of SRE12(c5) plus trials where $\mathcal{A}$ is clean. In $\mathcal{E}$, each target speaker has several enrollment sessions from several different recording conditions (we used the enrollment file list that excludes repeated speech). There is only one model ID for each speaker in $\mathcal{E}$. Therefore, #M is equal to #Es.

Table 5.1 shows the number of files that we had in $\mathcal{E}$ and $\mathcal{A}$ after discarding corrupted files. In $\mathcal{E}$ of SRE12, there were 8094 and 12393 files for training 723 and 1095 male and female speaker models, respectively. Since the PIN numbers were missing for unknown test segments, #As and #Us for male and female tasks of SRE12 could not be counted. Table 5.2 shows the number of trials in the evaluation sets. For all sets, only a small number of the non-target trials have impostors who are unseen in $\mathcal{E}$. In more realistic scenarios, the number of unseen impostors might be higher. In SRE12, this is taken into account by a re-balancing of the trials.

Note that, using $\mathcal{E}$ for system development is allowed in the NIST SRE plan for SRE12 but not for SRE06, SRE08 and SRE10. Therefore, we violated the rules of SRE06, SRE08 and SRE10.

5.4.2 Background Datasets ($\mathcal{R}$ and $\mathcal{C}$)

As the background dataset, we used the NIST SRE 2004 (SRE04), NIST SRE 2005 (SRE05), Switchboard II Phase 1 (SB2P1), Switchboard II Phase 2 (SB2P2), Switchboard II Phase 3 (SB2P3), Switchboard Cellular Part 1

Table 5.1: Development set, SRE06 and evaluation sets, SRE08, SRE10 and SRE12 for male and female speakers. #M: the number of models in the enrollment set, $\mathcal{E}$, #Es: the number of unique speakers in $\mathcal{E}$, #Te: the number of test files in the authentication set, $\mathcal{A}$, #As: the number of unique speakers in $\mathcal{A}$, and #Us: the number of speakers of $\mathcal{A}$ unseen in $\mathcal{E}$.

Dataset	Male				
	#M	#Es	#Te	#As	#Us
SRE06	349	257	1347	257	4
SRE08	648	492	858	427	105
SRE10	1906	187	384	192	20
SRE12(c2)	723	723	4962	-	-
SRE12(c4)	723	723	3900	-	-
SRE12(c5)	723	723	2156	-	-
Dataset	Female				
	#M	#Es	#Te	#As	#Us
SRE06	459	335	1679	327	5
SRE08	1140	844	1508	691	92
SRE10	2361	221	369	208	11
SRE12(c2)	1095	1095	7984	-	-
SRE12(c4)	1095	1095	6195	-	-
SRE12(c5)	1095	1095	3325	-	-

(SBCP1) and Switchboard Cellular Part 2 (SBCP2). From SRE04, we selected speech files having single-channel conversation of approximately five minutes total duration. From SRE05, we selected speech files having two-channel conversation of approximately five minutes total duration. We used all non-empty speech files of the Switchboard datasets. We prepared two background datasets, $\mathcal{R}$ and $\mathcal{C}$. For training UBMs and i-vector extractor, $\boldsymbol{T}$, we used $\mathcal{R}$, whereas for training PLDA models we used both $\mathcal{R}$ and $\mathcal{C}$.

Background dataset $\mathcal{R}$ included all speech files available in the above mentioned speech databases. The number of speech files, #F, and the number of speakers, #S, of individual dataset and of the combined dataset, $\mathcal{R}$, are shown in Table 5.3. MIXER PIN and PIN were used as unique speaker IDs for NIST SRE and Switchboard datasets, respectively. For the files whose

Table 5.2: Trials of SRE06, SRE08, SRE10 and SRE12 for male and female speakers. #T: the number of total trials, #Tr: the number of target trials, #Nt: the number of non-target trials, #Kn: the number of non-target trials by known speakers, #Un: the number of non-target trials by unknown speakers.

Dataset	Male				
	#T	#Tr	#Nt	#Kn	#Un
SRE06	22123	1594	20529	20066	463
SRE08	12356	724	11632	9906	1726
SRE10	179338	3465	175873	158846	17027
SRE12(c2)	164549	2830	161719	131932	29787
SRE12(c4)	125400	2775	122625	122625	0
SRE12(c5)	62845	1534	61311	61311	0
Dataset	Female				
	#T	#Tr	#Nt	#Kn	#Un
SRE06	28945	2022	26923	26478	445
SRE08	22957	1445	21512	20088	1424
SRE10	236781	3704	233077	221097	11980
SRE12(c2)	393042	4524	388518	313109	75409
SRE12(c4)	298491	4401	294090	289218	4872
SRE12(c5)	152976	2349	150627	148221	2406

MIXER PIN or PIN were missing, model IDs were used as speaker IDs. For example, in SRE05, there were 198 male and 211 female speech segments without MIXER PIN. We counted those speech segments as from 28 male and 29 female speakers based on their model IDs. However, multiple model IDs may share the same MIXER PIN. Therefore, it is possible that our counted #S was higher than the original #S. There were 18 male and 23 female speakers appearing in multiple Switchboard datasets. Therefore, the number of speakers in the combined set, $\mathcal{R}$, was smaller than the total number of speakers in individual sets. For male set, #S of $\mathcal{R}$ was 1495, whereas the total number of speakers was 1525. For female set, #S of $\mathcal{R}$ was 1897, whereas the total number of speakers was 1919.

For $\mathcal{C}$, we selected only the *clean speech* files of $\mathcal{R}$, i.e., $\mathcal{C} \subset \mathcal{R}$. Clean

Table 5.3: The number of speech files, #F, and the number of speakers, #S, used for training gender-dependent UBM and $\mathbf{T}$.

Dataset	Male		Female	
	#F	#S	#F	#S
SB2P1	2558	292	3251	358
SB2P2	2352	304	2716	335
SB2P3	1612	290	2083	341
SBCP1	462	103	567	116
SBCP2	1310	165	2000	245
SRE04	1906	126	2651	188
SRE05	2705	245	3792	336
$\mathcal{R}$	12905	1495	17060	1897

speech refers to speech which is not distorted by *echo* or *crosstalk* or *background noise* according to the meta-data of the databases. According to the documentation of the Switchboard corpora [48], echo or crosstalk in the telephone circuit refers to the audibility of the channel-1 speaker in channel-2 and vice-versa. Background noise refers to the amount of sounds not made by the speakers, e.g., baby crying, television, radio, etc. For the NIST SRE databases, there is no meta-data for identifying *noisy* speech, therefore we considered all speech of SRE04 and SRE05 as clean speech. The number of speakers, #S, and the number of clean speech files, #F, of individual dataset and of the combined dataset, $\mathcal{C}$, are given in Table 5.4.

5.4.3 Pre-processing and Training Models

We at first extracted 15 PLP coefficients [64] along with log-energy and then applied feature warping [106]. After that we appended the first-order and second-order derivatives, resulting in 48 elements per frame. Then we removed non-speech parts from the feature vector sequences by using spectral subtraction-based voice activity detector (VAD) [99].

After extracting PLP features, we trained gender-dependent systems using $\mathcal{R}$. First, we trained gender-dependent UBMs with 2048 Gaussian components by using feature vectors of $\mathcal{R}$. Next we trained gender-dependent $\mathbf{T}$ matrices by the feature vectors extracted from $\mathcal{R}$. The rank of $\mathbf{T}$ matrices,

Table 5.4: The number of speech files, #F, and the number of speakers, #S, selected from clean speech for training gender-dependent PLDA models.

Dataset	Male		Female	
	#F	#S	#F	#S
SB2P1	391	125	455	191
SB2P2	1868	283	2134	307
SB2P3	1399	277	1921	337
SBCP1	236	78	290	94
SBCP2	1038	157	1595	232
SRE04	1906	126	2651	188
SRE05	2705	245	3792	336
$\mathcal{C}$	9543	1278	12838	1665

d, was tuned to 400 by using SRE06. By using $\mathbf{T}$ matrices, we extracted i-vectors. Then, we applied data selection methods for selecting i-vectors for training PLDA models. Finally, the i-vectors of PLDA models went through the process of centering, whitening, and length-normalization [43].

We trained four gender-dependent PLDA models, among which two models were for male speakers and two models were for female speakers using $\mathcal{R}$ and $\mathcal{C}$. The parameters $\mathbf{m}$, $\mathbf{V}$ and $\mathbf{\Sigma}$ of PLDA models were estimated by the ML criteria (see Equation 3.52). The rank of $\mathbf{V}$ was optimized to 250 by using SRE06. Table 5.5 defines the symbols for referring to the data sets we used in the experiments. Note that we will use the same symbol for the training set and its corresponding PLDA model from now on.

5.4.4 Tuning k

For the conventional k-NN, we optimised k by minimising EER of the development set, SRE06. Using the cosine distance as the distance metric, we chose the k-nearest neighbours from $\mathcal{C}$ for each $\boldsymbol{\omega}_e \in \mathcal{E}$. We increased k from one up to fifty. When $k \leq 2$, PLDA training failed due to an insufficient amount of training data. The optimum k was 37 for male and 25 for female trials of SRE06, respectively. We used the same k for the background dataset, $\mathcal{R}$.

Table 5.5: Symbols that will be used for referring to PLDA models later in this paper.

Symbol	Training Data
$\mathcal{R}$	All available data
$\mathcal{C}$	Clean data selected by removing echo or crosstalk or noise from $\mathcal{R}$, i.e., $\{\mathcal{C} \subset R\}$
$\{\mathcal{C}/\mathcal{R}\}_{\text{k}}$	$\mathcal{S} \subset \{\mathcal{C}/\mathcal{R}\}$ selected by k-NN
$\{\mathcal{C}/\mathcal{R}\}_{\text{fk}}$	$\mathcal{S} \subset \{\mathcal{C}/\mathcal{R}\}$ selected by fk-NN
$\mathcal{C}_{\text{ik}}$	$\mathcal{S} \subset \mathcal{C}$ selected by ik-NN
$\{\mathcal{C}/\mathcal{R}\} + \mathcal{E}$	Training data added with enrolment set
$\{\mathcal{C}/\mathcal{R}\}_{\text{k}} + \mathcal{E}$	$\mathcal{S} \subset \{\mathcal{C}/\mathcal{R}\}$ selected by k-NN and added with $\mathcal{E}$
$\{\mathcal{C}/\mathcal{R}\}_{\text{fk}} + \mathcal{E}$	$\mathcal{S} \subset \{\mathcal{C}/\mathcal{R}\}$ selected by fk-NN and added with $\mathcal{E}$

5.4.5 Performance Measure

For SRE06, SRE08 and SRE10, we used *equal error rate* (EER) and *minimum detection cost*, C^{min}, as evaluation metrics. For SRE12, we used a minimum and an actual version of the primary evaluation metric, denoted by C^{min} and C^{act}, respectively, as evaluation metrics. C^{min} was the C_{avg} (See Section **??**). Both $C^{\beta_1}_{\text{Norm}}$ and $C^{\beta_2}_{\text{Norm}}$ used $C_{\text{FR}} = C_{\text{FA}} = 1$.

We computed C^{act} by applying detection thresholds of $\log(\beta)$ for the two values of β with $\beta_1 = 99$ and $\beta_2 = 999$ as recommended in NIST SRE plan for SRE12. We used an affine transformation estimated using the C_{llr} loss shifted to $P_{\text{tar}} = 10^{-2.5}$ (i.e., the geometric average of $P^{(1)}_{\text{tar}}$ and $P^{(2)}_{\text{tar}}$). We used SRE06 for training the affine transformation. For C^{min}, we applied a PAV transformation on the evaluation scores. For calculating compound LLRs, doing calibration and calculating the evaluation metrics, we used the BOSARIS toolbox [13].

5.5 Results

This section presents results of our experiments conducted on the development set, SRE06, and three evaluation sets, SRE08, SRE10 and SRE12. Section 5.5.1 shows results of SRE06,SRE08 and SRE10, whereas Section 5.5.2 shows results of SRE12.

5.5.1 SRE06, SRE08 and SRE10

Table 5.6 compares EER and C^{min} for the baseline, k-NN and fk-NN based ASVS. Data selection either by k-NN or by fk-NN improved the verification accuracy. The k-NN method performed well on the development set, SRE06, where k was optimized. On the other hand, fk-NN was better than k-NN for reducing EER in SRE08 and SRE10. Using fk-NN in $\mathcal{C}$, we achieved on average 4.2% and 3.4% relative reduction in EER over the baseline for male and female trials of the evaluation sets, respectively. On the other hand, using fk-NN in $\mathcal{R}$ we achieved on average 6.0% and 5.9% relative reduction in EER over the baseline for male and female trials of the evaluation sets, respectively.

We followed steps of the significance test mentioned in Section 2.7. By comparing fk-NN based system with baseline system using $\mathcal{R}$, we got p values 6.366e-05, 0.1167 and 0.0267 for male trials of SRE06, SRE08 and SRE10 respectively. For female trials of SRE06, SRE08 and SRE10, p- values were1.0051e-07, 0.000116, and 0.0197 respectively. We can say that except SRE08 male trials, our achievement using fk-NN was statistically significant at $\alpha = 0.05$ level. fk-NN was more successful for improving system's performance for non-target trials than target trials. In SRE08 male, the number of non-target trials was not large, which effect on the overall p value of SRE08 male.

5.5.2 SRE12

In order to explore whether k-NN and fk-NN are effective for noisy data, we performed experiments on SRE12. SRE12 is different from the previous evaluation sets in that several enrollment sessions are available for each speaker in $\mathcal{E}$. In our preliminary experiments, the methods using the averaged i-vectors, a-k-NN and a-fk-NN (Section 5.3.3), were always better

Table 5.6: EER and C^{min} of SRE06, SRE08 and SRE10. For SRE06 and SRE08, C^{min} is in 10^{-2} whereas for SRE10, C^{min} is in 10^{-4}. For all tasks EER is in %.

Male	SRE06		SRE08		SRE10	
model	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	2.30	1.16	4.92	2.55	2.01	3.73
$\mathcal{C}_{\text{k}}$	**1.84**	**1.05**	4.76	2.44	2.05	3.68
$\mathcal{C}_{\text{fk}}$	2.08	1.12	**4.73**	**2.43**	**1.92**	**3.53**
$\mathcal{R}$	2.59	1.33	5.07	2.65	2.14	3.97
$\mathcal{R}_{\text{k}}$	2.08	**1.13**	4.87	**2.58**	2.11	3.94
$\mathcal{R}_{\text{fk}}$	**2.07**	1.15	**4.77**	**2.58**	**2.01**	**3.76**
Female	SRE06		SRE08		SRE10	
model	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	3.42	1.85	5.97	2.85	3.02	4.94
$\mathcal{C}_{\text{k}}$	**2.71**	**1.43**	5.81	**2.82**	2.93	**4.74**
$\mathcal{C}_{\text{fk}}$	**2.71**	**1.43**	**5.78**	2.84	**2.91**	**4.74**
$\mathcal{R}$	3.92	2.20	6.29	3.01	3.29	4.96
$\mathcal{R}_{\text{k}}$	**2.89**	**1.50**	5.80	**2.84**	3.22	4.83
$\mathcal{R}_{\text{fk}}$	**2.89**	**1.50**	**5.79**	**2.84**	**3.16**	**4.81**

than the methods using all available i-vectors per speaker. Therefore, we considered only a-k-NN and a-fk-NN in these experiments. Further, the probability that an impostor is one of the other target speakers, P_{Known}, was considered in this evaluation. In this Section, we used $P_{\text{Known}} = 0.5$ which was the core task of the evaluation. The effect of changing the value of P_{Known} is analyzed in Section 5.6.6.

The results of data selection from $\mathcal{C}$ and $\mathcal{R}$ are given in Table 5.7 and Table 5.8. Since SRE12(c4) and SRE12(c5) for male do not have any unknown impostors, the performance for these conditions could not be estimated. With the exception of using $\mathcal{C}$ for male, data selection was always effective. In many cases, k-NN was better than fk-NN. A possible reason for this could be that averaged i-vectors were not optimal for determining k with fk-NN. It is noticeable that, despite the fact that $\mathcal{A}$ was noisy, using $\mathcal{C}$ gave in most cases better C^{act} than using $\mathcal{R}$. This could perhaps be explained by the fact that the calibration model was trained on SRE06 which

Table 5.7: Results of male trials of SRE12(c2) using $P_{Known} = 0.5$. In k-NN, k was optimized considering SRE06 as the development set.

PLDA model	C^{act}	C^{min}
$\mathcal{C}$	0.321	**0.293**
$\mathcal{C}_{a\text{-}k}$	**0.320**	0.298
$\mathcal{C}_{a\text{-}fk}$	0.325	0.315
$\mathcal{R}$	0.350	0.287
$\mathcal{R}_{a\text{-}k}$	0.331	0.289
$\mathcal{R}_{a\text{-}fk}$	**0.323**	**0.279**

Table 5.8: Results of female trials of SRE12 using $P_{Known} = 0.5$. In k-NN, k was optimized considering SRE06 as the development set.

PLDA	c2		c4		c5	
model	C^{act}	C^{min}	C^{act}	C^{min}	C^{act}	C^{min}
$\mathcal{C}$	0.432	0.281	0.598	0.464	0.491	0.310
$\mathcal{C}_{a\text{-}k}$	**0.386**	**0.271**	**0.548**	**0.447**	**0.436**	0.308
$\mathcal{C}_{a\text{-}fk}$	0.414	0.279	0.574	0.452	0.470	**0.307**
$\mathcal{R}$	0.476	0.281	0.630	0.444	0.536	0.317
$\mathcal{R}_{a\text{-}k}$	**0.409**	0.277	**0.558**	**0.441**	**0.461**	0.314
$\mathcal{R}_{a\text{-}fk}$	0.445	**0.273**	0.596	0.445	0.504	**0.299**

was clean.

5.6 Analysis

In this Section we analyze the behavior of data selection with k-NN and fk-NN more in details, as well as some of their modifications and extensions described in Section 5.3. Most of the experiments were done on SRE06, SRE08 and SRE10. Only for checking the effect of unseen impostors, SRE12 was used.

5.6.1 Analysis of the Selected Data

Fig. 5.6(a)-5.6(f) shows how much data were selected from each training corpus for SRE06, SRE08 and SRE10. The most noticeable trend was that

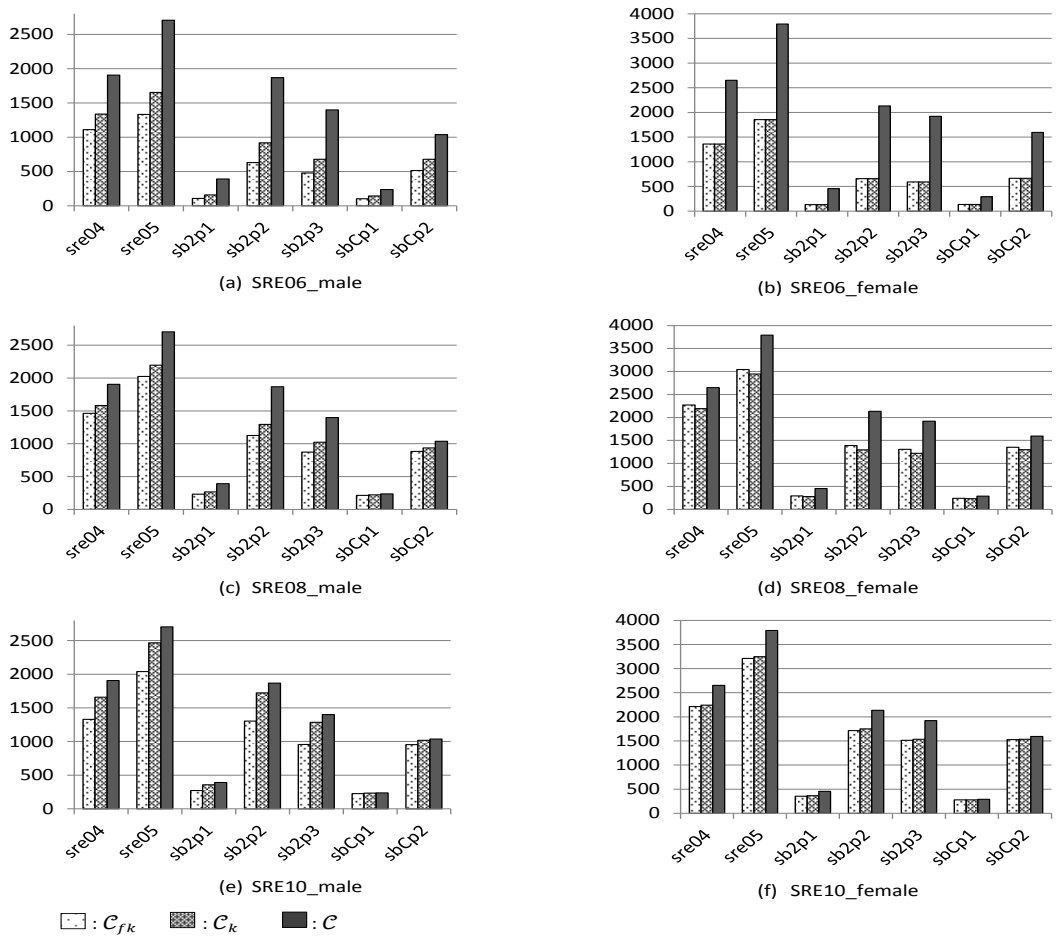

Figure 5.6: The y-axis shows the number of i-vectors of different datasets used for training PLDA models for male and female trials of SRE06, SRE08 and SRE10.

SRE10 selected much more of the Switchboard corpora than SRE06 and SRE08. In particular, SRE10 selected almost all of SBCP1. For further analysis, we trained database-specific PLDA models using data of each database in $\mathcal{C}$. For SB2P1 and SBCP1, PLDA training failed due to an insufficient amount of training data. In this thesis, we did not attempt to solve this problem by applying regularization to the channel covariance during PLDA training. The results are shown in Table 5.9. We can conclude that the NIST SRE databases (ALLSRE) have more relevant data for $\mathcal{E}$ of SRE06, SRE08, and SRE10 than the Switchboard databases (ALLSB). Using only ALLSRE, we got the lowest EER and C^{min} for SRE06 and SRE08 while adding ALLSB

with ALLSRE had negative impact on the performance. It reveals that using all the available data does not guarantee the best PLDA model for the target evaluation set. The presence of irrelevant data in the training set of the PLDA model may deteriorate the system's performance. For SRE10, we got the lowest EER and C^{min} when we combined ALLSB with ALLSRE. It indicates that relevant data differs in different target evaluation sets. By k-NN and fk-NN, we are able to reduce the amount of irrelevant data for the target evaluation set.

5.6.2 Error Analysis

In order to analyze the errors, we counted the number of *false acceptance* (FA) and *false rejection* (FR) as well as the number of enrollment and test segments that had at least one erroneous decision for any trial in SRE06, SRE08 and SRE10. For this analysis, we used the thresholds that minimized the detection costs. For the baseline system, the number of FR was higher than the number of FA. This is because the operating point of C^{min} in SRE06, SRE08 and SRE10 promotes a low FA rate. This is particularly extreme for SRE10. As shown in Table 5.10, we noticed that data selection reduced the number of FR, miss-verified speakers and test-segments in all datasets except SRE08. In most of the cases, the number of FA increased when data selection was applied.

5.6.3 Adding All Sessions from Selected Speakers

According to the data selection approach, k-NN-s, described in Section 5.3.3, all the sessions of each selected speaker were added in to $\mathcal{S}$ and k was optimized. The optimal value of k on SRE06 was 12 and 3 for male and female tasks, respectively, compared to 37 and 25 for the standard approach. Table 5.11 shows the results. As can be seen, this method did not perform well with the values of k that were optimal for the standard approach. However, when k was specifically optimized for this purpose, the result was comparable to the standard approach. This approach could, however, be refined by ensuring that every speaker has at least a certain number of sessions rather than using all the available sessions. Such exploration will be a part of future work.

Table 5.9: EER and C^{min} of SRE06, SRE08 and SRE10 for different training data sets. Empty entries mean that PLDA training failed due to insufficient amount of training data. For SRE06 and SRE08, C^{min} is in 10^{-2} whereas for SRE10, C^{min} is in 10^{-4}. For all tasks, EER is in %.

Male model	SRE06		SRE08		SRE10	
	EER	C^{min}	EER	C^{min}	EER	C^{min}
SB2P1	-	-	-	-	-	-
SB2P2	10.65	4.99	13.26	5.54	17.96	8.56
SB2P3	11.23	5.28	14.31	5.65	18.19	8.78
SBCP1	-	-	-	-	-	-
SBCP2	16.88	6.0	15.16	5.79	12.47	9.81
ALLSB	8.17	3.86	9.65	4.83	5.38	6.54
SRE04	4.89	2.20	6.96	3.29	4.88	7.51
SRE05	3.85	1.94	5.74	2.99	2.81	5.65
ALLSRE	**2.07**	**0.97**	**4.58**	**2.36**	2.28	4.20
$\mathcal{C}$	2.30	1.16	4.92	2.55	**2.01**	**3.73**
Female model	SRE06		SRE08		SRE10	
	EER	C^{min}	EER	C^{min}	EER	C^{min}
SB2P1	-	-	-	-	-	-
SB2P2	12.56	6.0	14.94	6.42	17.92	9.15
SB2P3	11.57	6.07	13.83	6.24	17.42	8.77
SBCP1	-	-	-	-	-	-
SBCP2	11.20	5.34	11.43	5.32	7.56	8.54
ALLSB	9.04	5.16	11.12	5.41	5.88	6.92
SRE04	3.35	1.73	6.53	3.00	4.62	6.83
SRE05	5.06	2.60	6.94	3.25	3.74	5.26
ALLSRE	**2.64**	**1.42**	**5.51**	**2.6**	3.26	**4.69**
$\mathcal{C}$	3.42	1.85	5.97	2.85	**3.02**	4.94

5.6.4 Domain Adaptation

Table 5.12 compares the baseline, k-NN and fk-NN when $\mathcal{E}$ was added to the PLDA training set. Notice that $\mathcal{E}$ was added after data selection. The addition of $\mathcal{E}$ improved the performance of all systems substantially. For male trials of SRE06, SRE08 and SRE10, by using $\mathcal{C}+\mathcal{E}$ we achieved $(2.30-$

Table 5.10: Number of errors. FR: False Rejection, FA : False Acceptance, eS: Erroneous Target Speakers, eT: Erroneous Test Segments.

SRE06	Male			Female		
	$\mathcal{C}$	$\mathcal{C}_{\text{k}}$	$\mathcal{C}_{\text{fk}}$	$\mathcal{C}$	$\mathcal{C}_{\text{k}}$	$\mathcal{C}_{\text{fk}}$
FR	137	113	114	244	209	209
FA	84	91	105	219	142	142
eS	124	112	117	226	194	194
eT	194	178	186	380	303	303
SRE08	Male			Female		
	$\mathcal{C}$	$\mathcal{C}_{\text{k}}$	$\mathcal{C}_{\text{fk}}$	$\mathcal{C}$	$\mathcal{C}_{\text{k}}$	$\mathcal{C}_{\text{fk}}$
FR	137	92	96	274	266	269
FA	104	163	155	265	269	267
eS	171	184	179	345	345	350
eT	196	194	186	405	405	404
SRE10	Male			Female		
	$\mathcal{C}$	$\mathcal{C}_{\text{k}}$	$\mathcal{C}_{\text{fk}}$	$\mathcal{C}$	$\mathcal{C}_{\text{k}}$	$\mathcal{C}_{\text{fk}}$
FR	1156	1098	1046	1529	1518	1438
FA	7	9	9	19	15	20
eS	798	768	737	1125	1118	1075
eT	253	252	246	290	281	276

$1.64)/2.30 \times 100\%$= 28.7%, 15.9% and 27.4% relative reduction in EER over $\mathcal{C}$, respectively. For female trials of SRE06, SRE08 and SRE10, the EER reduction rates were 29.5%, 14.7% and 16.9%, respectively. These results confirmed the effect of domain adaptation, i.e., adding $\mathcal{E}$ to $\mathcal{C}$.

We observed a consistent improvement using data-selection followed by domain adaptation. By using $\mathcal{C}_{\text{fk}} + \mathcal{E}$, we achieved 6.7% and 4.1% EER reduction over $\mathcal{C} + \mathcal{E}$ for male trials of SRE06 and SRE08, respectively. For female trials of SRE06, SRE08 and SRE10, the EER reduction rates were 13.3%, 4.7% and 6.8%, respectively. Fig. 5.7 shows the DET curves. It is clear that adding $\mathcal{E}$ to $\mathcal{C}$ improved PLDA modelling and that fk-NN improved the system performance further by discarding irrelevant data from $\mathcal{P}$.

When $\mathcal{E}$ was included with $\mathcal{R}$, using fk-NN, the EER reduction rates were 18.1%, 10.2% and 3.7%, respectively, for male trials of SRE06, SRE08

Table 5.11: EER and C^{min} for speaker based i-vector selection. k was tuned on SRE06. The "*" indicates that k was optimized for this method. In the other rows, k was optimized before adding discarded sessions of the selected speakers. For SRE06 and SRE08, C^{min} is in 10^{-2} whereas for SRE10, C^{min} is in 10^{-4}. For all tasks, EER is in %.

Male	SRE06		SRE08		SRE10	
model	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	2.30	1.16	4.92	2.55	**2.01**	**3.73**
$\mathcal{C}_{\text{k-s}}, k = 37$	2.28	1.21	5.01	2.58	2.08	3.93
$\mathcal{C}^*_{\text{k-s}}, k = 12$	**2.03**	**1.12**	**4.86**	**2.48**	2.04	3.77
$\mathcal{C}_{\text{fk-s}}$	2.22	1.19	4.89	2.54	2.08	3.88
Female	SRE06		SRE08		SRE10	
model	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	3.42	1.85	5.97	2.85	**3.02**	4.94
$\mathcal{C}_{\text{k-s}}, k = 25$	3.50	1.96	6.20	2.98	3.19	4.95
$\mathcal{C}^*_{\text{k-s}}, k = 3$	**2.71**	**1.43**	**5.79**	**2.82**	3.05	**4.75**
$\mathcal{C}_{\text{fk-s}}$	3.50	1.96	6.17	2.99	3.20	4.90

and SRE10. For female trials of SRE06, SRE08 and SRE10, the EER reduction rates were 23.8%, 9.1% and 4.9%, respectively. We can conclude that $\{\mathcal{P}_{\text{fk}} + \mathcal{E}$ or $\mathcal{P}_{\text{k}} + \mathcal{E}\} > \{\mathcal{P} + \mathcal{E}\} >> \{\mathcal{P}_{\text{fk}}$ or $\mathcal{P}_{\text{k}}\} > \{\mathcal{P}\}$, where $>$ refers *better* and $>>$ refers *much better* performance.

5.6.5 Individual k-NN

In all of our experiments up until now, we used the same k for all $\omega_e \in \mathcal{E}$. Here, we show experiment with ik-NN proposed in Section 5.3.3. Table 5.13 shows results of using $\gamma = 0.0001$. Overall, ik-NN outperformed our baselines systems, but it was not better than fk-NN. A comparison of this method and the standard k-NN for different amounts of training data is shown in Figure 5.8. For k-NN, the amount of training data was controlled by varying the value of k, and for ik-NN the amount of training data was controlled by varying the threshold, γ. For smaller training data sizes, ik-NN was better but for larger sizes, k-NN was better. Both the methods reached, however, a similar optimum. There is therefore no clear winner of fk-NN and ik-NN.

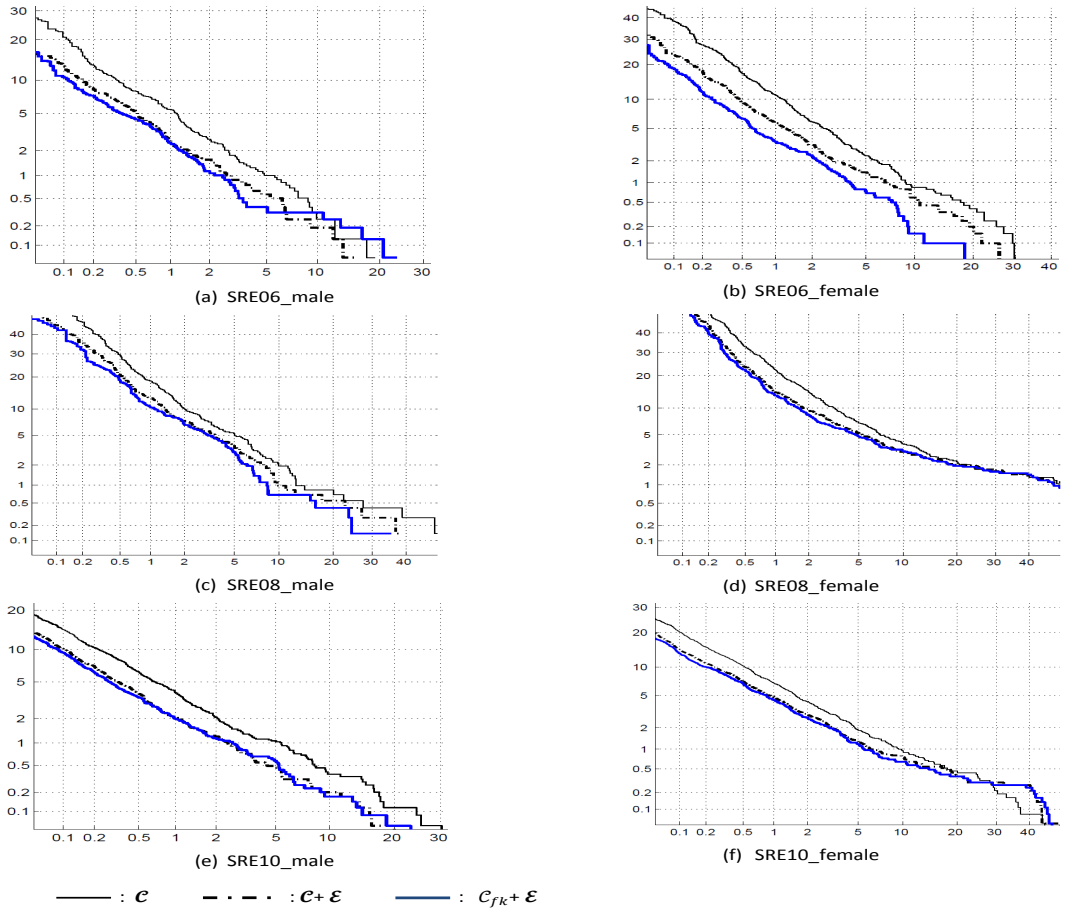

Figure 5.7: DET curves comparison of PLDA models trained by using different amount of data. The results are given for male and female trials of SRE06, SRE08 and SRE10. The x-axis shows *False Alarm Probability (in %)* and the y-axis shows *Miss Probability (in %)*.

Using ω_e dependent k is however tricky and the proposed ik-NN is unlikely to be the best approach. Exploring other strategies may, therefore, be a fruitful direction of future work.

5.6.6 Effect on Unseen Impostors

As discussed in Section 5.3.4, we need to confirm whether data selection has a bad effect on unknown impostors. For this, we examined the performance of k-NN and fk-NN on SRE12(c2) when $\mathrm{P}_{\mathrm{Known}} = 1$ and $\mathrm{P}_{\mathrm{Known}} = 0$. The results are given in Table 5.14. Overall, the performance of all methods

Table 5.12: EER and C^{min} for systems trained by including $\mathcal{E}$ into $\mathcal{P}$. In k-NN, k was optimised using SRE06. For male, $k = 37$, and for female, $k = 25$. For SRE06 and SRE08, C^{min} is in 10^{-2} whereas for SRE10, C^{min} is in 10^{-4}. For all tasks EER is in %.

Male	SRE06		SRE08		SRE10	
model	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	2.30	1.16	4.92	2.55	2.01	3.73
$\mathcal{C} + \mathcal{E}$	1.64	0.90	4.14	2.00	1.46	2.97
$\mathcal{C}_{\text{k}} + \mathcal{E}$	**1.35**	**0.76**	**3.92**	1.92	1.48	3.05
$\mathcal{C}_{\text{fk}} + \mathcal{E}$	1.53	0.77	3.97	**1.80**	**1.46**	**2.91**
$\mathcal{R}$	2.59	1.33	5.07	2.65	2.14	3.97
$\mathcal{R} + \mathcal{E}$	1.88	1.01	4.41	2.27	1.61	3.21
$\mathcal{R}_{\text{k}} + \mathcal{E}$	**1.52**	**0.81**	4.04	**2.06**	**1.55**	3.26
$\mathcal{R}_{\text{fk}} + \mathcal{E}$	1.54	**0.81**	**3.96**	2.08	**1.55**	**2.95**
Female	SRE06		SRE08		SRE10	
model	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	3.42	1.85	5.97	2.85	3.02	4.94
$\mathcal{C} + \mathcal{E}$	2.41	1.29	5.09	2.21	2.51	4.31
$\mathcal{C}_{\text{k}} + \mathcal{E}$	**2.09**	**1.01**	**4.74**	2.12	2.36	4.06
$\mathcal{C}_{\text{fk}} + \mathcal{E}$	**2.09**	**1.01**	4.85	**2.08**	**2.34**	**4.04**
$\mathcal{R}$	3.92	2.20	6.29	3.01	3.29	4.96
$\mathcal{R} + \mathcal{E}$	2.90	1.47	5.49	2.42	2.68	4.42
$\mathcal{R}_{\text{k}} + \mathcal{E}$	**2.21**	**1.03**	**4.96**	**2.21**	2.57	**4.14**
$\mathcal{R}_{\text{fk}} + \mathcal{E}$	**2.21**	**1.03**	4.99	2.23	**2.55**	4.16

became better when $P_{\text{Known}} = 1$, since we used compound LLRs that took advantage of the presence of known impostors. When $P_{\text{Known}} = 0$, data selection resulted in much improvement for female but a less clear pattern for male. However, notice in Table 5.2 that the number of trials from unknown impostors is quite small for male, so these results might be less reliable. In conclusion, it does not seem unknown non-target trials become problematic if our data selection methods are used.

Table 5.13: EER and C^{min} for systems trained by $\mathcal{C}$, and $\mathcal{C}_{ik}$. For both male and female, $\gamma = 0.0001$. For SRE06 and SRE08, C^{min} is in 10^{-2} whereas for SRE10, C^{min} is in 10^{-4}. For all tasks EER is in %.

Male model	SRE06		SRE08		SRE10	
	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	2.30	1.16	4.92	2.55	2.01	3.73
$\mathcal{C}_k$	**1.84**	**1.05**	4.76	**2.44**	2.05	3.68
$\mathcal{C}_{fk}$	2.08	1.12	4.73	2.43	**1.92**	**3.53**
$\mathcal{C}_{ik}$	1.86	1.12	**4.54**	2.46	2.00	3.72
Female model	SRE06		SRE08		SRE10	
	EER	C^{min}	EER	C^{min}	EER	C^{min}
$\mathcal{C}$	3.42	1.85	5.97	2.85	3.02	4.94
$\mathcal{C}_k$	**2.71**	**1.43**	5.81	**2.82**	2.93	**4.74**
$\mathcal{C}_{fk}$	**2.71**	**1.43**	**5.78**	2.84	**2.91**	**4.74**
$\mathcal{C}_{ik}$	2.93	1.46	5.83	2.84	2.93	4.77

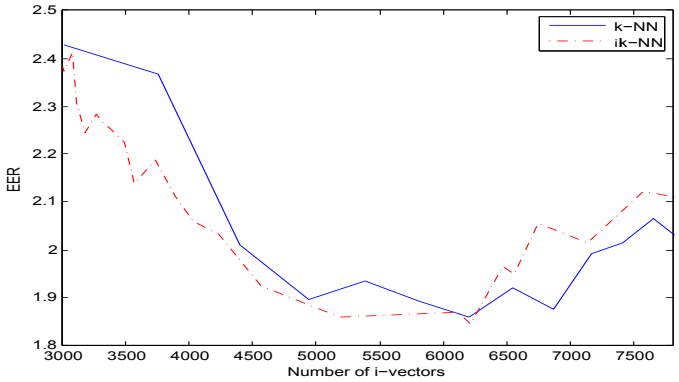

Figure 5.8: EER(%) of SRE06, male. The x-axis shows the number of i-vectors selected by k-NN and ik-NN for training the PLDA model.

5.6.7 Data Reduction Rate

Table 5.15 shows the data reduction rates for the four data sets. It is clear that more irrelevant data was reduced from $\mathcal{R}$ than $\mathcal{C}$ by both k-NN and fk-NN. For the male sets, fk-NN reduced the data more than k-NN. For SRE10, both k-NN and fk-NN discarded only a few speakers. For female, k-NN reduced more data than fk-NN in most cases.

Table 5.14: Results of SRE12(c2) using $\mathrm{P}_{\mathrm{Known}} = 1$ and $\mathrm{P}_{\mathrm{Known}} = 0$. For male, $k = 37$, and for female, $k = 25$.

Male model	$\mathrm{P}_{\mathrm{Known}} = 1$		$\mathrm{P}_{\mathrm{Known}} = 0$	
	$\mathrm{C}^{\mathrm{act}}$	$\mathrm{C}^{\mathrm{min}}$	$\mathrm{C}^{\mathrm{act}}$	$\mathrm{C}^{\mathrm{min}}$
$\mathcal{C}$	0.340	**0.246**	0.341	0.340
$\mathcal{C}_{\text{a-k}}$	0.339	0.257	**0.327**	**0.305**
$\mathcal{C}_{\text{a-fk}}$	**0.336**	0.268	0.363	0.342
$\mathcal{R}$	0.348	**0.246**	**0.330**	0.337
$\mathcal{R}_{\text{a-k}}$	0.340	0.254	0.347	0.326
$\mathcal{R}_{\text{a-fk}}$	**0.334**	0.257	0.338	**0.314**
Female model	$\mathrm{P}_{\mathrm{Known}} = 1$		$\mathrm{P}_{\mathrm{Known}} = 0$	
	$\mathrm{C}^{\mathrm{act}}$	$\mathrm{C}^{\mathrm{min}}$	$\mathrm{C}^{\mathrm{act}}$	$\mathrm{C}^{\mathrm{min}}$
$\mathcal{C}$	0.414	**0.239**	0.433	0.339
$\mathcal{C}_{\text{a-k}}$	0.395	**0.239**	**0.380**	0.341
$\mathcal{C}_{\text{a-fk}}$	**0.406**	0.240	0.414	**0.326**
$\mathcal{R}$	0.446	0.235	0.493	0.322
$\mathcal{R}_{\text{a-k}}$	**0.405**	0.235	**0.402**	**0.317**
$\mathcal{R}_{\text{a-fk}}$	0.424	**0.230**	0.452	0.322

Table 5.15: Data reduction rate (%) by k-NN and fk-NN. In k-NN, k was optimized using SRE06. For male, $k = 37$, and for female, $k = 25$. M: Male model, F: Female model, m: reduction rate (%) of i-vectors and n: reduction rate (%) of speakers. For SRE12, the results refer to a-fk-NN and a-k-NN.

M	SRE06		SRE08		SRE10		SRE12	
	m	n	m	n	m	n	m	n
$\mathcal{C}_{\text{k}}$	41.7	11.1	21.3	4.5	8.3	0.6	22.7	4.6
$\mathcal{C}_{\text{fk}}$	55.2	19.5	28.6	7.1	25.8	4.2	31.8	7.7
$\mathcal{R}_{\text{k}}$	50.9	10.6	30.7	4.3	12.9	0.8	31.5	4.0
$\mathcal{R}_{\text{fk}}$	52.7	11.5	33.2	4.8	26.2	3.1	43.5	7.5
F	SRE06		SRE08		SRE10		SRE12	
	m	n	m	n	m	n	m	n
$\mathcal{C}_{\text{k}}$	57.9	20.2	26.3	6.3	14.7	2.8	30.9	8.2
$\mathcal{C}_{\text{fk}}$	57.9	20.2	23.0	5.8	15.7	3.0	11.5	1.9
$\mathcal{R}_{\text{k}}$	65.5	17.9	34.9	5.1	20.6	1.8	39.7	6.3
$\mathcal{R}_{\text{fk}}$	65.5	17.9	31.7	4.1	19.6	1.7	17.1	1.7

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In the last two decades, short utterances and inter-session variability have been considered two important obstacles for achieving high verification accuracy for a text-independent automatic speaker verification (ASVS). In this thesis, we presented methods in order to deal with these obstacles and improve the performance of a text-independent ASVS. We focused on robust speaker modeling by improving background modeling.

We showed that SMAP adaptation is better than the popular MAP adaptation for reliable parameter estimation of speaker-specific models when utterances are very short (e.g., 10 seconds). In SMAP adaptation, a single tree structure of the UBM is generally used for the acoustic space of all the speakers assuming that the hierarchical structure of the acoustic space can be shared among all the speakers. During our work on speaker verification, however, we noticed that a single tree structure is not always optimal for modeling the acoustic space of every speaker. Therefore, for short utterances, we propose to grow an acoustic forest with different tree structures for SMAP adaption, and combine the decision of several SMAP adapted systems using score fusion techniques. By doing experiment on the 10sec4w-10sec4w task of the NIST SRE 2006, we gained 1.5% relative reduction EER.

We also presented data selection methods for PLDA modeling, which is one of the state-of-the-art methods for i-vector scoring. Using k-NN we showed that we can choose a subset of the available training data of the

PLDA model, and improve the system performance for both male and female trials of the NIST SRE 2006, SRE 2008, SRE 2010, and SRE 2012. In order to avoid the difficulty of optimizing k on a development set, we presented a robust way of selecting k, named *flexible* k-NN (fk-NN), which uses a *local distance-based outlier factor* (LDOF). This method discarded irrelevant or noisy training data of the PLDA model as much as the conventional k-NN without the need for tuning k. Using both k-NN and fk-NN, we achieved reduced EER and detection costs. By fk-NN, we reduced on average 30% irrelevant data and achieved 6.0% reduction in EER for male 2.5 minutes tasks of SRE 2008 and SRE 2010. For the female task, our achievement were 26% irrelevant data reduction and 6.6% reduction in EER. We also proposed variations of these methods, including *individual* k-NN (ik-NN) which uses different k for different data points. The effect of i-vector selection on known and unknown non-target trials, was also a topic of this thesis.

Relation Between Our Proposed Methods

Both SMAP tree and PLDA model provide prior information for estimating robust parameters of speaker-specific model. SMAP tree is concerned about data sparseness problem. It does not deal with inter-session variability. Its success is quite dependent on the human decided pre-specified tree structure and the quality of data used for building the UBM. On the other hand, a PLDA model provides prior information in order to separate inter-session variability factors from speaker-specific factors. Its success depends on the quality of the training data. Our proposed methods, acoustic forest and fk-NN, increase robustness of these two models. Acoustic forest increases robustness of an SMAP tree against enrollment data variants and fk-NN increases robustness of a PLDA model against irrelevant and noisy training data.

Robustness against enrollment data variants, and robustness against irrelevant and noisy training data are not two completely separate issues for background modeling. One approach can deal with both issues and improve the same background model. Our proposed two approaches can be used for the same background model as proposed in [124, 125, 94]. In SMAP tree we need to train a UBM at first. We can use fk-NN to select relevant data for

UBM training. In that case, instead of using i-vectors, we need to use feature vectors like MFCC, PLPCC etc. In an i-vector based ASVS, one i-vector is estimated from a complete utterance. On the other hand, multiple feature vectors are extracted from an utterance. For example, from a 2.56 minutes long utterance we can extract 17598 MFCCs in total if we extract one MFCC from 30ms speech per 10ms. From a 10 seconds utterance we can extract only 998 MFCCs. In order to minimize the computational expense, we need to make one single vector from n feature vectors before applying fk-NN. We can take averaged of n feature vectors or we can do polynomial expansion of the averaged feature vector and make a single vector per utterance. We can then apply fk-NN to select data for UBM training. After training UBM, we can make acoustic forest by clustering Gaussian components of that UBM applying different tree structures.

It is also possible to use acoustic forest in i-vector based ASVS. Structured adaptation for JFA has been proposed in [36] and the same method can be applied to other factor analysis models such as total variability models. Accordingly, acoustic forest and fk-NN could be therefore combined.

6.2 Future Work

For the acoustic forest based system, we only gave a comparative figure of relevance MAP and SMAP for short speech segments. In future work, we plan to compare SMAP adaptation with other adaptation techniques, such as eigenvoice modeling.

For fk-NN, future directions are many. It would be interesting to see whether the performance of gender-dependent PLDA models can be improved by selecting data from the opposite gender. Our proposed data selection methods do not depend on any channel compensation techniques. Therefore, it would be a good idea to explore whether they can benefit from methods such as WCCN, NAP or LDA. We should also explore how much training data is required for training an efficient PLDA model. Further developments of ik-NN, as well of schemes for adding discarded i-vectors from the selected speakers seem to be promising directions. Also, the current method uses the unique set of the selected i-vectors, and thus ignores the number of times the i-vectors have been selected. Taking this information into account could be interesting.

We should also explore the effect of data selection by k-NN and fk-NN in GMM-supervector space. Its success may help us in reducing training time of the total variability matrix. However, in [6], it has been argued that as the dimensionality increases, the distance to the nearest neighbor approaches the distance to the farthest neighbour. This holds true for a broad range of distributions and distance measures including cosine similarity measure [109]. Therefore, both k-NN and fk-NN using the cosine similarity or cosine distance may become ill-defined for high dimensional supervectors. Therefore, we need to explore other distance metrics.

Publications

- Journal Paper

    1. **Sangeeta Biswas**, Johan Rohdin, Koichi Shinoda, Autonomous Selection of i-Vectors for PLDA Modelling in Speaker Verification, Speech Communication, vol. 72, pp. 32-46, Sep. 2015.

- Conference & Workshop Proceedings (peer reviewed)

    1. **Sangeeta Biswas**, Johan Rohdin, Koichi Shinoda, i-Vector Selection for Effective PLDA Modeling in Speaker Recognition, Odyssey 2014: The Speaker and Language Recognition Workshop, pp. 100-105, Jun. 16, 2014.
    2. **Sangeeta Biswas**, Marc Ferras, Koichi Shinoda, Sadaoki Furui, Acoustic Forest for SMAP-based Speaker Verification, INTERSPEECH, pp. 2377-2380, Aug. 27, 2011.

Bibliography

[1] C. C. Aggarwal and P. S. Yu. Outlier Detection for High Dimensional Data. In *ACM International Conference on Management of Data (SIGMOD)*, pages 37–46, 2001.

[2] E. Alpaydin. Voting over Multiple Condensed Nearest Neighbors. *Artificial Intelligence Review*, 11:115–132, 1997.

[3] B. S. Atal. Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. *Journal of the Acoustical Society of America*, 55(6):1304–1312, 1974.

[4] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas. Score Normalization for Text-independent Speaker Verification Systems. *Digital Signal Processing*, 10:42–54, 2000.

[5] S. Bengio and J. Mariéthoz. A Statistical Significance Test for Person Authentication. In *Odyssey 2004: The Speaker and Language Recognition Workshop*, pages 237–244, 2004.

[6] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When Is "Nearest Neighbor" Meaningful? In *International Conference on Database Theory (ICDT)*, pages 217–235, 1999.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, USA, 2006.

[8] S. Biswas, M. Ferras, K. Shinoda, and S. Furui. Acoustic Forest for SMAP-based Speaker Verification. In *INTERSPEECH*, pages 2377–2380, 2011.

[9] S. Biswas, J. Rohdin, and K. Shinoda. i-Vector Selection for Effective PLDA Modeling in Speaker Recognition. In *Odyssey 2014: The Speaker and Language Recognition Workshop*, pages 100–105, 2014.

[10] S. Biswas, J. Rohdin, and K. Shinoda. Autonomous selection of i-vectors for PLDA modelling in speaker verification. *Speech Communication*, 72:32–46, 2015.

[11] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[12] M. M. Bruce. Estimation of variance by a recursive equation. *NASA Technical note*, 1969.

[13] N. Brümmer. SRE'12 - BOSARIS Toolkit. In *https://sites.google.com/site/bosaristoolkit/sre12*, 2012.

[14] N. Brümmer and E. de Villiers. The speaker partitioning problem. In *Odyssey 2010: The Speaker and Language Recognition Workshop*, pages 194–201, 2010.

[15] L. Burget, P. Matějka, P. Schwarz, O. Glembek, and J. Černocký. Analysis of Feature Extraction and Channel Compensation in GMM Speaker Recognition System. *IEEE Transactions on Audio, Speech & Language Processing*, 15(7):1979–1986, 2007.

[16] W. M. Campbell. Generalized Linear Discriminant Sequence Kernels for Speaker Recognition. In *ICASSP*, volume 1, pages I – 161–164, 2002.

[17] W. M. Campbell, J. R. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek. High-level speaker verification with support vector machines. In *ICASSP*, volume 1, pages I – 73–76, 2004.

[18] W. M. Campbell, D. E. Sturim, and D. A. Reynolds. Support Vector Machines using GMM Supervectors for Speaker Verification. *IEEE Signal Processing Letters*, 13(5):308–311, 2006.

[19] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff. SVM Based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation. In *ICASSP*, pages I – 97–100, 2006.

[20] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[21] P. E. Cheng. Strong Consistency of Nearest Neighbor Regression Function Estimators. *Journal of Multivariate Analysis*, 15:63–72, 1984.

[22] T. M. Cover. Rates of Convergence for Nearest Neighbor Procedures. In *Hawaii International Conference on System Sciences*, 1968.

[23] T. M. Cover and P. E. Hart. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[24] S. Davis and P. Mermelstein. Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Speech and Audio Processing*, 28(4):357–366, 1980.

[25] N. Dehak and G. Chollet. Support Vector GMMs for Speaker Verification. *Odyssey 2006: The Speaker and Language Recognition Workshop*, pages 1–4, 2006.

[26] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet, and P. Dumouchel. Support Vector Machines versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification. In *INTERSPEECH*, pages 1559–1562, 2009.

[27] N. Dehak, R. Dehak, P. Kenny, and P. Dumouchel. Comparison Between Factor Analysis and GMM Support Vector Machines for Speaker Verification. In *Odyssey 2008: The Speaker and Language Recognition Workshop*, 2008.

[28] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-End Factor Analysis for Speaker Verification. *IEEE Transactions on Audio, Speech & Language Processing*, 19(4):788–798, 2011.

[29] L. Devroye and T. J. Wagner. Nearest Neighbor Methods in Discrimination. *Handbook of Statistics 2: Classification, Pattern Recognition and Reduction of Dimensionality (P. R. Krishnaiah and L. N. Kanal, eds.)*, pages 193–197, 1982.

[30] L. P. Devroye and T. J. Wagner. The Strong Uniform Consistency of Nearest Neighbor Density Estimates. *The Annals of Statistics*, 5(3):536–540, 1977.

[31] C. Domeniconi, J. Peng, and D. Gunopulos. Locally Adaptive Metric Nearest-Neighbor Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285, 2002.

[32] G. G. Enas and S. C. Choi. Choice of the smoothing parameter and efficiency of k-nearest neighbor classification. *Computers & Mathematics with Applications*, 12A(2):235–244, 1986.

[33] X. Fan and J. H. L. Hansen. Speaker Identification Within Whispered Speech Audio Streams. In *IEEE Transactions on Audio, Speech and Language Processing*, volume 19, pages 1408–1421, 2011.

[34] B. Fauve, N. Evans, and J. Mason. Improving the performance of text-independent short duration SVM- and GMM-based speaker verification. In *Odyssey2008: The Speaker and Language Recognition Workshop*, 2008.

[35] B. Fauve, N. Evans, N. Pearson, J.-F. Bonastre, and J. Mason. Influence of task duration in text-independent speaker verification. In *INTERSPEECH*, pages 794–797, 2007.

[36] Marc Ferras, Koichi Shinoda, and Sadaoki Furui. Structural Joint Factor Analysis for Speaker Recognition. In *INTERSPEECH*, pages 2373–2376, 2011.

[37] E. Fix and Jr. J. L. Hodges. Discriminatory Analysis - Nonparametric Discrimination: Consistency Properties. In *Randolph Field, Texas, Project 21-49-004, Report No. 4.*, 1951.

[38] H. Frank. *Introduction to Probability and Statistics: Concepts and Principles*. John Wiley and Sons, Inc, 1974.

[39] K. Fukunaga and L. Hostetler. Optimization of k-Nearest Neighbor Density Estimates. *IEEE Transactions on Information Theory*, 19(3):320–326, 1973.

[40] K. Fukunaga and L. Hostetler. k-Nearest-Neighbor Bayes-Risk Estimation. *IEEE Transactions on Information Theory*, 21(3):285–293, 1975.

[41] S. Furui. Cepstral analysis technique for automatic speaker verification. *IEEE Transactions on Acoustics, Speech, and Signal processing*, 29(2):254–272, 1981.

[42] S. Furui. Speaker Independent Isolated Word Recognition Using Dynamic Features on Speech Spectrum. *IEEE Transactions on Acoustics, Speech, and Signal processing*, 34(1):52–59, 1986.

[43] D. Garcia-Romero and C. Y. Espy-Wilson. Analysis of i-vector Length Normalization in Speaker Recognition Systems. In *INTERSPEECH*, pages 249–252, 2011.

[44] G. Gates. The Reduced Nearest Neighbour Rule. *IEEE Transactions on Information Theory*, 18:431–433, 1972.

[45] J. L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE Transactions on Speech and Audio Processing*, 2:291–298, 1994.

[46] A. K. Ghosh. On optimum choice of k in nearest neighbor classification. *Computational Statistics and Data Analysis*, 50(11), 2006.

[47] Ondřej Glembek, Lukáš Burget, Najim Dehak, Niko Brümmer, and Patrick Kenny. Comparison of Scoring Methods used in Speaker Recognition with Joint Factor Analysis. In *ICASSP*, 2009.

[48] D. Graff, A. Canavan, and G. Zipperlen. Switchboard-2 Phase I. In *Linguistic Data Consortium, Philadelphia*, 1998.

[49] C. S. Greenberg, D. Bansé, G. R. Doddington, D. Garcia-Romero, J. J. Godfrey, T. Kinnunen, A. F. Martin, A. McCree, M. Przybocki, and D. A. Reynolds. The NIST 2014 Speaker Recognition i-Vector Machine Learning Challenge. In *Odyssey 2014: The Speaker and Language Recognition Workshop*, pages 224–230, 2014.

[50] L. Györfi. The rate of convergence of k-nn regression estimates and classification rules. *IEEE Transactions on Information Theory*, 27:362–364, 1981.

[51] L. Györfi and Z. Györfi. An upper bound on the asymptotic error probability of the k-nearest neighbor rule for multiple classes. *IEEE Transactions on Information Theory*, 24:512–514, 1978.

[52] P. Hall, B. U. Park, and R. J. Samworth. Choice of neighbor order in nearest-neighbor classification. *The Annals of Statistics, Institute of Mathematical Statistics*, 36(5):2135–2152, 2008.

[53] E.-H. S. Han, G. Karypis, and V. Kumar. Text categorization using weight adjusted k -nearest neighbor classification. In *5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 53–65, 2001.

[54] J. H. L. Hansen. Analysis and compensation of speech under stress and noise for environmental robustness in speech recognition. *Speech Communication*, 20(1–2):151–173, 1996.

[55] P. Hart. The Condensed Nearest Neighbour Rule. *IEEE Transactions on Information Theory*, 14:515–416, 1968.

[56] T. Hasan and J. H. L. Hansen. A Study on Universal Background Model Training in Speaker Verification. *IEEE Transactions on Audio, Speech & Language Processing*, 19(7):1890–1899, 2011.

[57] T. Hasan, Y. Lei, A. Chandrasekaran, and J. H. L. Hansen. A novel feature sub-sampling method for efficient universal background model training in speaker verification. In *ICASSP*, pages 4494–4497, 2010.

[58] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):606–616, 1996.

[59] A. O. Hatch, S. S. Kajarekar, and A. Stolcke. Within-class covariance normalization for SVM-based speaker recognition. In *INTERSPEECH*, 2006.

[60] A. O. Hatch and A. Stolcke. Generalized linear kernels for one-versus-all classification: application to speaker recognition. In *ICASSP*, 2006.

[61] D. Hawkins. Identification of outliers. *Chapman and Hall*, 1980.

[62] T. J. Hazen. A comparison of novel techniques for rapid speaker adaptation. *Speech Communications*, 31:15–33, 2000.

[63] L. P. Heck and M. Weintraub. Handset-dependent background models for robust text-independent speaker recognition. In *ICASSP*, pages 1071–1074, 1997.

[64] H. Hermansky. Perceptual predictive (PLP) analysis of speech. *The Acoustical Society of America*, 87(4):1738–1752, 1990.

[65] H. Hermansky, N. Morgan, A. Bayya, , and P. Kohn. Rasta-plp speech analysis technique. In *ICASSP*, pages I.121–I.124, 1992.

[66] M. Holst and A. Irle. Nearest neighbor classification with dependent training sequences. *The Annals of Statistics*, 29:1424–1442, 2001.

[67] R. Hsiao and B. Mak. Kernel eigenspace-based MLLR adaptation using multiple regression classes. In *ICASSP*, volume 1, pages 985–988, 2005.

[68] C.-L. Huang and B. Ma. Maximum Entropy Based Data Selection for Speaker Recognition. In *INTERSPEECH*, pages 2713–2716, 2011.

[69] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *the 30th Annual ACM Symposium on Theory of Computing, STOC'98*, pages 604–613, 1998.

[70] S. Ioffe. Probabilistic Linear Discriminant Analysis. In *ECCV (4)*, pages 531–542, 2006.

[71] A. K. Jain, A. Ross, and S. Prabhakar. An Introduction to Biometric Recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20, 2004.

[72] H. M. Wang K. T. Chen, W. W. Liau and L. S. Lee. Fast speaker adaptation using eigenspace-based maximum likelihood linear regression. 3:742–745, 2000.

[73] S. Kajarekar. Four weightings and a fusion: a cepstral-svm system for speaker recognition. In *ASRU*, 2005.

[74] A. Kanagasundaram, D. Dean, J. Gonzalez-Dominguez, S. Sridharan, D. Ramos, and J. Gonzalez-Rodriguez. Improving Short Utterance based I-vector Speaker Recognition using Source and Utterance-Duration Normalization Techniques. In *INTERSPEECH*, pages 2465–2469, 2013.

[75] A. Kanagasundaram, D. Dean, S. Sridharan, M. McLaren, and Robbie Vogt. I-vector based speaker recognition using advanced channel compensation techniques. *Computer Speech and Language*, 28:121–140, 2014.

[76] A. Kanagasundaram, R. Vogt, D. Dean, S. Sridharan, and M. Mason. i-vector Based Speaker Recognition on Short Utterances. In *INTERSPEECH*, pages 2341–2344, 2011.

[77] A. Kanagasundaram, R. J. Vogt, D. B. Dean, and S. Sridharan. PLDA based Speaker Recognition on Short Utterances. In *Odyssey 2012: The Speaker and Language Recognition Workshop*, pages 28–33, 2012.

[78] P. Kenny. Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms, Tech. Report CRIM-06/08-13. *[Online]. Available: http://www.crim.ca/perso/patrick.kenny/*, 2005.

[79] P. Kenny. Bayesian speaker verification with heavy-tailed priors. In *Odyssey 2010: The Speaker and Language Recognition Workshop*, 2010.

[80] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. Joint Factor Analysis Versus Eigenchannels in Speaker Recognition. *IEEE Transactions on Audio, Speech & Language Processing*, 15(4):1435–1447, 2007.

[81] P. Kenny and N. Dehak. Robust Speaker Recognition Over Varying Channels. In *Report from JHU workshop*, pages 20–42, 2008.

[82] P. Kenny and P. Demouchel. Eigenvoices modeling with sparse training data. *IEEE Transactions on Speech and Audio Processing*, 13(3):345–354, 2005.

[83] P. Kenny and P. Dumouchel. Disentangling speaker and channel effects in speaker verification. In *ICASSP*, volume 1, pages 47–50, 2004.

[84] P. Kenny, T. Stafylakis, P. Ouellet, M. Alam, and P. Dumouchel. Plda for speaker verification with utterances of arbitrary duration. In *ICASSP*, 2013.

[85] T. Kinnunen and H. Li. A Overview of Text-independent Speaker Recognition: From Features to Supervectors. *Speech Communication*, 52:12–40, 2010.

[86] E. Knorr and R. Ng. Finding Intensional Knowledge of Distance-Based Outliers. In *VLDB Conference*, 1999.

[87] M. Kubat and Jr. M. Cooperson. Voting Nearest-Neighbour Subclassifiers. In *Proceedings of the 17th International Conference on Machine Learning, ICML-2000*, pages 503–510, 2000.

[88] R. Kuhn, J. Junqua, P. Ngyuen, and N. Niedzielski. Rapid speaker adaptation in eigenvoice space. *IEEE Transactions on Speech and Audio Processing*, 8(6):695–707, 2000.

[89] S. R. Kulkarni and S. E. Posner. Rates of convergence of nearest neighbor estimation under arbitrary sampling. *IEEE Transactions on Information Theory*, 41:1028–1039, 1995.

[90] N. Kumar. Investigation of Silicon Auditory Models and Generalization of Linear Discriminant Analysis for Improved Speech Recognition. *Ph.D. Dissertation, Johns Hopkins University, Baltimore, Maryland*, 1997.

[91] Q. Le and S. Bengio. Client Dependent GMM-SVM Models for Speaker Verification. *ICANN/ICONIP, Lecture Notes in Computer Science*, pages 443–451, 2003.

[92] C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(2):171–185, 1995.

[93] K. P. Li, J. E. Dammann, and W. D. Chapman. Experimental studies in speaker verification using an adaptive system. *The Journal of the Acoustical Society of America*, 40:966–978, 1966.

[94] G. Liu, J.-W. Suh, and J. H. L. Hansen. A Fast Speaker Verification with Universal Background Support Data Selection. In *ICASSP*, pages 4793–4796, 2012.

[95] M. Liu, E. Chang, and B. Q. Dai. Hierarchical Gaussian Mixture Model for Speaker Verification. In *ICSLP*, 2002.

[96] S. Lucey and T. Chen. Improved speaker verification through probabilistic subspace adaptation. In *Eurospeech*, pages 2021–2024, 2003.

[97] J. Luck. Automatic speaker verification using cepstral measurements. *The Journal of the Acoustical Society of America*, 46:1026–1031, 1969.

[98] B. Mak and R. Hsiao. Improving eigenspace-based mllr adaptation by kernel pca. In *ICSLP*, pages 13–16, 2004.

[99] M. W. Mak and H. B. Yu. Robust voice activity detection for interview speech in NIST speaker recognition evaluation. In *APSIPA ASC*, 2010.

[100] M.W. Mak, R. Hsiao, and B. Mak. A comparison of various adaptation methods for speaker verification with limited enrollment data. In *ICASSP*, pages 929–932, 2006.

[101] M. McLaren, B. Baker, R. Vogt, and S. Sridharan. Data-driven Background Dataset Selection for SVM-based Speaker Verification. *IEEE Transactions on Audio, Speech and Language Processing*, 18(6):1496–1506, 2010.

[102] M. McLaren, R. Vogt, B. Baker, and S. Sridharan. Data-driven Impostor Selection for T-norm Score Normalisation and the Background Dataset in SVM-based Speaker Verification. *Advances in Biometrics*, 5558(7):474–483, 2009.

[103] M. McLaren, R. Vogt, B. Baker, and S. Sridharan. Experiments in SVM-based Speaker Verification Using Short Utterances. In *Odyssey 2010: The Speaker and Language Recognition Workshop*, pages 83–90, 2010.

[104] NIST. The NIST Year 2005 Speaker Recognition Evaluation Plan. In *http://www.itl.nist.gov/iad/mig/tests/spk/2005/index.html*, 2005.

[105] NIST. The NIST Year 2006 Speaker Recognition Evaluation Plan. In *http://www.itl.nist.gov/iad/mig/tests/spk/2006/index.html*, 2006.

[106] J. Pelecanos and S. Sridharan. Feature Warping for Robust Speaker Verification. In *2001: A Speaker Odyssey - The Speaker Recognition Workshop*, pages 213–218, 2001.

[107] S. J. D. Prince and J. H. Elder. Probabilistic Linear Discriminant Analysis for Inferences About Identity. *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[108] S. Pruzansky. Pattern-matching procedure for automatic talker recognition. *The Journal of the Acoustical Society of America*, 35:354–358, 1963.

[109] M. Radovanovic, A. Nanopoulos, and M. Ivanovic. On the Existence of Obstinate Results in Vector Space Models. In *SIGIR*, pages 186–193, 2010.

[110] D. Reynolds. Speaker identification and verification using Gaussian mixture speaker models. *Speech Communication*, 17(1–2):91–108, 1995.

[111] D. Reynolds, M. Zissman, T. Quatieri, G. O'Leary, and B. Carlson. The effects of telephone transmission degradations on speaker recognition performance. *ICASSP*, pages 329–332, 1995.

[112] D. A. Reynolds. The effect of handset variability on speaker recognition performance: experiments on the switchboard corpus. In *ICASSP*, volume 1, pages 113–116, 1996.

[113] D. A. Reynolds. Channel robust speaker verification via feature mapping. In *ICASSP*, volume 2, pages 53–56, 2003.

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

[115] J. Rohdin, S. Biswas, and K. Shinoda. Discriminative PLDA training with application-specific loss functions in speaker verification. *Computer Speech and Language*, 35:32–57, 2016.

[116] A. K. Sarkar and S. Umesh. Investigation of Speaker-Clustered UBMs based on Vocal Tract Lengths and MLLR matrices for Speaker Verification. In *Odyssey 2010: The Speaker and Language Recognition Workshop*, pages 286–293, 2010.

[117] M. A. Schuh, T. Wylie, and R. A. Angryk. Improving the performance of high-dimensional knn retrieval through localized dataspace segmentation and hybrid indexing. In *the 17th ADBIS Conference*.

[118] M. Senoussaoui, P. Kenny, N. Brümmer, E. de Villiers, and P. Dumouchel. Mixture of PLDA Models in I-Vector Space for Gender-Independent Speaker Recognition. In *INTERSPEECH*, pages 25–28, 2011.

[119] K. Shinoda and C.-H. Lee. A Structural Bayes Approach to Speaker Adaptation. *IEEE Transactions on Speech and Audio Processing*, 9(3):276–287, 2001.

[120] S. Shum, N. Dehak, E. Chuangsuwanich, D. A. Reynolds, and J. R. Glass. Acoustic Forest for SMAP-based Speaker Verification. In *INTERSPEECH*, pages 945–948, 2011.

[121] R. R. Snapp and S. S. Venkatesh. Asymptotic expansion of the k nearest neighbor risk. *The Annals of Statistics*, 26:850–878, 1998.

[122] Y. Song, J. Huang, D. Zhou, H. Zha, and C. L. Giles. IKNN: Informative K-nearest Neighbor Pattern Classification. In *PKDD*, 2007.

[123] K. Stevens, C. Williams, J. Carbonell, and B. Woods. Speaker authentication and identification: a comparison of spectrographic and auditory presentation of speech material. *The Journal of the Acoustical Society of America*, 44:596–607, 1968.

[124] D. E. Sturim and D. A. Reynolds. Speaker Adaptive Cohort Selection for Tnorm in Text-Independent Speaker Verification. In *ICASSP*, pages 741–744, 2005.

[125] J.-W. Suh, Y. Lei, W. Kim, and J. H. L. Hansen. Effective Background Data Selection in SVM Speaker Recognition for Unseen Test Environment: More is Not Always Better. In *ICASSP*, pages 5304–5307, 2011.

[126] R. Teunen, B. Shahshahani, and L. Heck. A model-based transformational approach to robust speaker recognition. In *International Conference on Spoken Language Processing*, 2000.

[127] S. van Vuuren and H. Hermansky. On the importance of components of the modulation spectrum for speaker verification. In *ICSLP*, volume 7, pages 3205–3208, 1998.

[128] V. N. Vapnik. The Nature of Statistical Learning Theory. *Springer-Verlag, New-York, NY, USA*, 1995.

[129] R. Vogt, B. Baker, and S. Sridharan. Factor Analysis Subspace Estimation for Speaker Verification with Short Utterances. In *INTERSPEECH*, 2008.

[130] R. Vogt, C. Lustri, and S. Sridharan. Factor Analysis Modelling for Speaker Verification with Short Utterances. In *Odyssey 2008: The Speaker and Language Recognition Workshop*, 2008.

[131] V. Wan and S. Renals. Svmsvm: Support vector machine speaker verification methodology. In *ICASSP*, volume 2, pages 221–4224, 2003.

[132] H. Wang. Nearest Neighbours without k: A Classification Formalism based on Probability. In *Technical report, Faculty of Informatics, University of Ulster, N. Ireland, UK*, 2002.

[133] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, 2005.

[134] D. R. Wilson and T. R. Martinez. Reduction Techniques for Exemplar-Based Learning Algorithms. *Machine learning*, 38(3):257–286, 2000.

[135] J. J. Wolf. Efficient acoustic parameters for speaker recognition. *The Acoustical Society of America*, 51(6):2044–2056, 1972.

[136] B. Xiang and T. Berger. Efficient Text-Independent Speaker Verification with Structural Gaussian Mixture Models and Neural Network. *IEEE Transactions on Speech and Audio Processing*, 11:447–456, 2003.

[137] Lan Yi. Eliminating Noisy Information in Web Pages for Data Mining. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 296–305, 2003.

[138] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book*. Cambridge University Engineering Department, 2002.

[139] C. Yu, B. C. Ooi, K.-L. Tan, and H. V. Jagadish. Indexing the Distance: An Efficient Method to KNN Processing. In *the 27th VLDB Conference*, pages 421–430, 2001.

[140] K. Zhang, M. Hutter, and H. Jin. A New Local Distance-Based Outlier Detection Approach for Scattered Real-World Data. In *PAKDD*, pages 813–822, 2009.

[141] S. X. Zhang and M. W. Mak. High-level speaker verification via articulatory-feature based sequence kernels and SVM. In *INTERSPEECH*, pages 1393–1396, 2008.

[142] W.-Q. Zhang, Y. Shan, and J. Liu. Multiple Background Models for Speaker Verification. In *Odyssey 2010: The Speaker and Language Recognition Workshop*, pages 47–51, 2010.