EFFECTS OF DATA DURATION, MODEL SIZE AND SESSION VARIABILITY ON SPEAKER VERIFICATION PERFORMANCE

by

Erinç Dikici

B.S., Telecommunication Engineering, İstanbul Technical University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2009

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Assist. Prof. Murat Saraçlar, for his invaluable guidance and help throughout this study. I have always admired his insightful approach to problems at hand. I appreciate his intimate attitude, patience and confidence in my ability to succeed.

I am grateful to my thesis committee members, Prof. Levent Arslan and Assist. Prof. Hakan Erdoğan, for their valuable ideas and encouraging interest in my studies from the very beginning of my graduate education. I also would like to thank Prof. Bülent Sankur for his outside help and support, and for broadening my horizons with his enlightening knowledge as well as his intellectual point of view.

My valuable thanks go to all my past and present colleagues at BUSIM for their friendship, for sharing my troubles during the preparation of this thesis, and for turning the lab into a warm environment with their presence.

I owe special thanks to Oya Çeliktutan, Sıddıka Parlak Polatkan, Ebru Arısoy, Ceyhun Burak Akgül, Cemil Demir, Özgür Devrim Orman, Gönenç Seçil Tarakcıoğlu, Ali Haznedaroğlu and Osman Büyük for their stimulating ideas, constructive comments and technical help. I would also like to thank friends at CmpE for their friendship and support, and for giving me the privilege of being a member of their cheerful family.

I would like to thank TUBITAK for supporting the financial means of my graduate study. This work has been supported by TUBITAK Grant No: 107E001 and is covered by the COST 2101 project "Biometrics for Identity Documents and Smart Cards".

Last but not least, I would like to express my deepest gratitude to my dear family, who have supported me with their endless love and understanding throughout my life.

ABSTRACT

EFFECTS OF DATA DURATION, MODEL SIZE AND SESSION VARIABILITY ON SPEAKER VERIFICATION PERFORMANCE

Speaker verification is one of the most challenging branches of biometric authentication. Covering a wide spectrum from security services to law enforcement, speaker verification systems are employed in phone banking, forensic audio analysis and access control applications. An important observation is that verification accuracies depend vastly on the amount of data and get easily affected by acoustic variations. This study investigates the effects of data duration, model size and session variability on text-independent speaker verification performance.

We implement GMM/UBM and SVM supervector classifiers to represent speaker characteristics and compare their results for various training and testing durations as well as model complexities. The influence of speaker adaptation methods and kernel function selection over the verification accuracy is examined. A minority oversampling scheme is utilized in order to avoid the issue of class imbalance in SVMs. We also explore how session variability acts on error rates and resort to Nuisance Attribute Projection method for reducing acoustic mismatches between the training and test samples. Working on the CSLU Speaker Recognition Dataset, we present a comparative evaluation of speaker verification systems with limited and extensive data conditions.

ÖZET

VERİ SÜRESİ, MODEL BÜYÜKLÜĞÜ VE OTURUM DEĞİŞKENLİĞİNİN KONUŞMACI DOĞRULAMA BAŞARIMINA ETKİSİ

Konuşmacı doğrulama, biyometrik kimlik denetiminin en zorlayıcı dallarından biridir. Güvenlik sistemlerinden yasal yürütüme kadar geniş bir yelpazede değerlendirilen konuşmacı doğrulama yöntemleri; telefon bankacılığı, adli ses çözümleme ve erişim kontrolü gibi alanlarda kullanılmaktadır. Bu uygulamalarda doğrulama başarımının veri miktarına önemli ölçüde bağlı olduğu ve ses kayıtlarındaki akustik değişimlerden kolayca etkilenebildiği gözlenmiştir. Bu çalışmada veri süresinin, model büyüklüğünün ve oturumlar arası değişkenliğin metinden bağımsız konuşmacı doğrulama başarımına etkisi incelenmektedir.

Konuşmacı karakteristiğini tanımlamada Gauss Karışım Modeli/Genel Arkaplan Modeli (GKM/GAM) ve buradan elde edilen süpervektörler ile oluşturulan Destek Vektör Makinaları (DVM) kullanılmış, değişken eğitim ve sınama uzunluklarına ve model karmaşıklıklarına göre sonuçlar karşılaştırılmıştır. Konuşmacı uyarlama yöntemlerinin ve çekirdek fonksiyonu seçiminin doğrulama başarımı üzerindeki etkisi araştırılmıştır. DVM'deki sınıf dengesizliğini gidermek için bir azınlık üst örnekleme yaklaşımı değerlendirilmiştir. Eğitim ve sınama örnekleri arasındaki uyumsuzluktan kaynaklanan oturumlar arası değişkenliğin hata oranlarını artırmasını önlemek amacıyla Sıkıntı Öznitelik İzdüşümü yöntemine başvurulmuştur. CSLU Konuşmacı Doğrulama Veri Kümesi üzerinde, gerek sınırlı gerekse kapsamlı veri durumları için konuşmacı doğrulama sistemlerinin karşılaştırılmalı değerlendirmesi sunulmaktadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

$\mathbf{A}$	MLLR adaptation regression matrix
b	SVM separating hyperplane shift
$\mathbf{b}$	MLLR adaptation bias vector
C	Number of classes
C	SVM misclassification cost
$\mathbf{C}$	Correlation matrix of $\mathbf{M}$
d	d-th vector element
D	Dimension of vector
$E_k[\cdot]$	Expected value at k-th mixture component
$f(\cdot)$	Logistic function
H	Number of sessions
$\mathbf{I}$	Identity matrix
k	k-th mixture component
K	Total number of mixture components
$K(\cdot,\cdot)$	Kernel function
$KL(\cdot\|\|\cdot)$	Kullback-Leibler divergence
$L(\cdot)$	Distance metric
$\mathbf{m}^a$	Supervector of utterance a
$\mathbf{m}_{h_j}^{s_i}$	Supervector of i-th speaker in j-th session
$\mathbf{M}$	Intersession variation matrix
n_k	Number of samples in k-th mixture component
p	Degree of polynomial kernel
$p_k(\cdot)$	Probability at k-th mixture component
$p(\cdot)$	SVM probability score
P_{FA}	False alarm probability
P_M	Miss probability
$P(\cdot)$	Projection operator
R	Reduced channel subspace matrix
s	Score

S	Number of speakers
$\mathbf{w}$	Normal of the SVM separating hyperplane
$\mathbf{x}$	Acoustic feature vector
$\mathbf{x}_i$	SVM input vector
$\mathbf{x}_t$	Acoustic vector at time t
$\mathbf{X}$	Utterance X
$\mathbf{X}_I$	Pseudo-impostor utterance
y_i	SVM class label
α_i	SVM dual form Lagrange multipliers
α_k	Adaptation coefficient for the k-th mixture component
β	Fusion weight
Δ	First-order difference for MFCC vector
λ^a	Model for utterance a
λ^{s_i}	Model for i-th speaker
λ^I	Pseudo-impostor model
λ_k	Model parameters for k-th mixture component
$\Lambda(\cdot)$	Likelihood score
$\widetilde{\Lambda}(\cdot)$	Normalized likelihood score
μ	Mean vector
ξ_i	SVM slack variable
σ	RBF kernel influence area
σ	Standard deviation
$\mathbf{\Sigma}$	Covariance matrix
τ	Likelihood ratio threshold
τ'	Log-likelihood ratio threshold
τ	Relevance factor
$\phi(\cdot)$	Nonlinear mapping
ω_k	Weight of the k-th mixture component

AT-Norm	Adaptive Test Normalization
CMS	Cepstral Mean Subtraction

CSLU	Center for Spoken Language Understanding
DCT	Discrete Cosine Transform
DET	Detection Error Tradeoff
DTW	Dynamic Time Warping
EER	Equal Error Rate
EM	Expectation Maximization
FA	Factor Analysis
GLDS	Generalized Linear Discriminant Sequence
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
H-Norm	Handset Normalization
KEMLLR	Kernel Eigenspace-Based MLLR
KL	Kullback-Leibler
LDC	Linguistic Data Consortium
LPC	Linear Predictive Coding
MAP	Maximum A-Posteriori
MFCC	Mel-Frequency Cepstral Coefficient
minDCF	Minimum Detection Cost Function
M-Norm	Model Normalization
MLLR	Maximum Likelihood Linear Regression
MPEG	Moving Pictures Experts Group
NAP	Nuisance Attribute Projection
NASE	Normalized Audio Spectrum Envelope
NIST	National Institute of Standards and Technology
NN	Nearest Neighbors
OAA	One-Against-All
OAO	One-Against-One
OGI	Oregon Graduate Institute
PCA	Principal Component Analysis
PIN	Personal Identification Number
RBF	Radial Basis Function

ROC	Receiver Operating Characteristic
RSW	Reference Speaker Weighting
SFA	Symmetrical Factor Analysis
SMOTE	Synthetic Minority Oversampling Technique
SRE	Speaker Recognition Evaluation
STFT	Short Time Fourier Transform
SVM	Support Vector Machine
T-Norm	Test Normalization
UBM	Universal Background Model
VQ	Vector Quantization
WCCN	Within Class Covariance Normalization
Z-Norm	Zero Normalization

1. INTRODUCTION

Person authentication has recently been a research area of increasing interest with the need for automatic access control in today's security and safety systems. Technological developments in audio and visual microelectronic devices, data storage environments and high-speed computing have further increased the demand for commercial and practical identification and authentication applications.

Conventional authentication methods are based on either something the person has (such as ID cards and badges), or something the person knows (such as passwords and PINs). However, such items can potentially be stolen, forged or deciphered. Furthermore, there is the risk of losing and forgetting. Biometric authentication methods try to overcome these risks by identifying the person by a unique characteristic of his/her own, instead of what he/she knows or possesses [1].

Biometric features are attributes which define the person's physiological characteristics or behavioral aspects. Physiological characteristics include the face, fingerprint, palm geometry, iris/retina patterns, and DNA of the individual; whereas behavioral aspects contain the signature, keystroke and gait (walking style). Voice (speech) is a valuable biometric which combines physiological and behavioral traits, as it is an outcome of both the person's vocal tract shape and his/her speaking style.

Unlike retina/iris or fingerprint scans which people may find bothering to provide, speech is a convenient and natural form of input. This specialty makes it one of the most compelling biometrics [2]. Since it does not require any additional equipment other than a microphone, it is also a cost-effective solution. These two properties further emphasize the importance of speech in remote (telephone-based or online) authentication applications.

Besides these advantages, speech also shows a challenging nature for several reasons: First, the characteristics of voice get affected by physical changes like aging,

illness, fatigue and stress. Second, microphone and transmission conditions, together with background noise might modify speech signals quite easily [3]. Finally, speaker recognition methods have vulnerability to voice transformation and mimicry [4].

Systems which use speech as a biometric measure cover a wide spectrum, from authentication applications to law enforcement. In telephone banking and e-commerce, voice can be used as a verifier to access customer accounts or as an indicator of a conscious transaction. Speaker identification can be used for indexing broadcast news programs or annotating recorded meetings, so that it becomes possible to spot the time intervals between which a particular speaker is speaking [5]. Speech biometrics has also an important role in forensic science, by providing a mathematical measure to identify or verify the individual under investigation, by analyzing audio recordings [6]. Last but not least, speaker verification allows an easy and uninterrupted way for access control to facilities and objects, and can often be consistently combined with other biometric features (multimodal authentication) to provide a higher level of security.

The performance of speaker verification methods greatly depends upon three main factors: Data duration, model complexity and acoustic conditions of the recordings. One of the most important factors is the amount of speech used to enroll the speakers to the system, a process which is called "training". The longer the duration of training data, the higher the verification performance. A similar rule also applies to testing, the stage of deciding upon an identity. Large amounts of speech data may be available in forensic and broadcast indexing applications. However, for a realistic security system access scenario, especially the testing duration (the time the speaker has to spend to get access) should be kept as short as possible. The technique chosen towards the solution of the problem, and the model size is also an important issue. Generally, more complex models are needed to get a similar performance with increasing amounts of data. One other big challenge is what is called the "intersession (or, session) variability", i.e., the acoustic mismatch between recordings of the training and testing sessions.

This study focuses on text-independent speaker verification and aims to investigate the performance of several methods with variable data durations, model complexi-

ties and acoustic conditions. The strengths and weaknesses of two modeling techniques, GMM/UBM and SVM, are examined. Considering the broad range of applications, experiments are performed both with limited data and extensive data conditions, taking also into account the corresponding model complexities. The effect of session variability and channel degradation on verification accuracy is reviewed by working on a multisession telephone speech database. We comment on the optimal selection and combination of system parameters, and construction of the training setup to obtain the highest performance under fixed test conditions.

The thesis is organized as follows: In Section 2, a brief introduction and a historical background on speaker verification is presented. Section 3 describes theoretical details of the application. In Section 4, we introduce our implementation and baseline experimental results, and touch upon some important issues. Behavior, and techniques to enhance the performance of the system with limited and extensive data conditions are emphasized in Section 5 and Section 6, respectively. Finally, a summary of the study, future directions and concluding remarks are given in Section 7.

2. SPEAKER VERIFICATION

2.1. Introduction to Speaker Recognition

Speaker recognition is the process of automatically recognizing the speaking person from a recording, using the characteristic information included in speech. There are two subproblems in speaker recognition: (i) Speaker identification, which involves determining the identity of a given speech segment; (ii) Speaker verification, which deals with deciding whether the speech belongs to a claimed identity. The former is a 1:N classification problem, whereas the latter gives a 1/0 decision. Having this binary nature, speaker verification is also called "speaker authentication", or "speaker detection".

2.2. Classification of Speaker Recognition Systems

Speaker recognition systems are classified in terms of several criteria. Based on whether test utterances are allowed to come from any unknown identity, speaker identification can be divided into closed or open set types. In closed set identification, the computer is sure that the test utterance belongs to one of the speakers (classes) it is trained with, therefore forces the system to decide on an identity. Open set identification adds "none of the above" option to the decision, so that the computer may reject to assign the utterance to any of the speakers, by comparing the decision score to a predefined threshold. Speaker verification systems, by definition, are open set setups.

Speaker recognition systems can also be classified according to whether the transcription of speech is known: In text-dependent recognition, the system knows beforehand what the speaker is going to say (i.e., the password). Since speech signal carries information on not only the speaking style of the speaker but also the utterance itself, text-dependent recognition achieves the highest performance. On the other hand, it is easy to deceive a text-dependent recognition system by recording the target speaker's

voice in advance and playing it back. To overcome this problem of cheating, text-prompted speaker recognition systems are developed, which instantaneously generates (or selects) a random word or sequence of numbers and asks the user to repeat it in a short amount of time. Finally in text-independent recognition what is being said is not known. Therefore it offers a more flexible system as well as a more challenging problem.

2.3. History and Literature Review

The history of speaker recognition dates back to 1970s. Since then, recognition systems have evolved in terms of both model complexity and widespread usability. Type of recordings has also grown from small databases having clean, controlled speech to large, realistic ones recorded in uncontrolled environments.

The earliest verification applications used pattern matching approaches such as dynamic time warping (DTW), vector quantization (VQ) and nearest neighbors (NN). With the increased use of GMMs in 1990s, it became possible to build probabilistic methods for text-independent speaker recognition. Meanwhile, HMMs took over the title of the leading text-dependent recognition method, by encoding the temporal evolution and statistical variation of the features in the probabilistic framework [2, 7].

Inspired by the pioneer study by Reynolds and Rose in 1995 [8], GMMs continue to be the most widely utilized modeling method for text-independent speaker verification. Many of the publications either use GMMs as a baseline reference to compare other classifiers to, or try to enhance its performance by the techniques they propose. Another successful classifier, SVM, has shown dominance on state-of-the-art methods since 2000.

The studies in [8] and [9] investigate the influence of GMM model size on system performance. In [10], Liu et al. suggest that for GMM/UBM, using only top 1 mixture for calculating the likelihood ratio yields comparable results with traditional score calculation. Dehak and Chollet [11] reformulate the GMM likelihood score in terms of

Kullback-Leibler (KL) divergence. Building their GMM baseline system on the ALIZE toolkit [12], Fauve et al. suggest a way to select an optimal MFCC feature dimension.

A variety of input features have been proposed for SVM-based speaker verification. Older approaches use directly the acoustic vectors as inputs to the SVM space [13]. In [14], log-likelihoods obtained in the GMM/UBM decision are used as a two-dimensional feature vector. A similar approach is named "speaker location" in [15]. An interesting idea to represent variable length utterances in an SVM setup is the GMM supervector approach, which uses stacked means of mixture components and can be thought as a mapping from a variable length utterance to a high fixed dimensional vector [16].

In order to allow the discriminant classifier to observe a sequence of data, several feature and kernel methods have been proposed. For example, Wan and Renals [17] use the Fisher kernel to map an arbitrary length sequence to a fixed length feature vector. Moreno and Ho propose a new SVM methodology that uses a kernel based on the KL divergence between generative models (GMMs) and that nearly halves the equal error rate of that obtained by using the Fisher kernel [18]. By applying the method on a more challenging dataset, [11] affirms its promising improvement, on the contrary to Louradour and Daoudi [19], who argue that this method cannot create robust results when the test sequences have short duration. Campbell et al. [20] propose a Global Linear Discriminant Sequence (GLDS) kernel, which is simply based on an explicit mapping of each sequence to a single vector in a feature space, using polynomial expansions [21]. The GMM supervector approach is also used with a variety of kernels, such as the GMM supervector linear kernel [16, 22, 23, 24], GMM L^2 inner product kernel [22], and the nonlinear kernel [11, 23]. In a latest study by Dehak et al. [25], the effect of combining three kernel matrices with linear weights to form an optimal kernel is investigated.

Many methods have been proposed to compensate for acoustic mismatch between training and testing data, namely, score (speaker and session) variability. These can be broadly classified into three groups as feature-based, score-based and model-based

compensation methods.

Feature-based methods include cepstral mean subtraction (CMS), variance normalization and feature mapping. These have the advantage that they can be applied to any speaker modeling technique [26].

Score-based normalization covers techniques to normalize the decision scores by using some pseudo-impostor data. T-Norm, H-Norm, Z-Norm are some examples to score normalization. Test Normalization (T-Norm) by Auckenthaler et al. [27] has been adopted by many of the references cited and also exploited in this study.

[28] mentions two methods to obtain T-Normed SVM output scores. For each T-Normed speaker, a binary SVM is trained with either the speaker against a background speaker corpus, or a background speaker against other samples of background. It has been shown that it is better to have no normalization than to apply the former T-Norm method, and that the latter method performs slightly better than baseline.

[29] applies a variant of T-Norm which uses speaker-dependent cohort speakers and call the strategy "adaptive T-Norm" (AT-Norm). [11] and [23] present another normalization technique, effective in non linear kernel systems called Model Normalization (M-Norm).

Two model-based channel compensation methods have been quite popularly used with SVM classifiers: Nuisance Attribute Projection (NAP) and Factor Analysis (FA). First proposed by [30], NAP aims to remove the dimensions which are irrelevant (which correspond to the channel component) from the SVM input space. This idea is further extended in [26]. FA, introduced by Kenny [31] and Vogt [32] decomposes the feature vector into a speaker-session-independent component, a speaker-dependent component and a session-dependent component. Vogt and Sridharan in [32] observe that at least 10 seconds of speech is required to estimate the session factors sufficiently. In a GMM supervector setup, NAP aims to remove the channel subspace by projection, whereas for FA, an iterative method is used to estimate the latent variables which are then

subtracted from the GMM supervector [16, 33]. The study by Fauve et al. [24] contains comparative results of NAP and FA applied on the same dataset and propose an alternative approach called Symmetrical Factor Analysis (SFA). Having thought that applying NAP in SVMs with nonlinear kernels would incur expensive computational costs in higher dimensional space, [34] proposes a way to apply NAP for nonlinear kernels by using kernel PCA. The Within-Class Covariance Normalization (WCCN) technique, which inversely weights the contribution of each direction based on their extent of variability, has been applied in combination with NAP in [15].

A great deal of studies concentrate on fusing GMM and SVM classifier outputs. In [28], two methods have been used for this purpose. The first one uses the linear combination, where T-Normed output scores are linearly combined with an appropriate weight factor. The second approach uses a single-layer perceptron neural network to fuse the scores. With equal T-Norm fusion weights, linear fusion behaves exactly as perceptron, and decreases EER by 2%. Linear logistic regression is another popular method for combining different scores [35]. [25] compares its kernel-level combination results with score-level fusion, using the naive Bayes and an optimal linear formulation having weights determined by logistic regression.

There are also studies to obtain more meaningful output scores. For example, in [14], the log-likelihood ratio has been changed with a linear regression model to obtain an optimal decision score. The parameters of the model are learned using the SVM classifier.

Another popular topic in speaker verification is to use higher level features which provide additional information on the speaking style of the speaker, especially where a significant amount of data is present. These include lexical features (differences of speakers' personal lexicon), phonetic features (personal variations in pronunciation and tendency to vocalize a variety of phones in similar ways), prosodic features (inter-personal variations in pitch and volume patterns) and durational features (variations present in the rate with which individuals produce different phones). In [36], it is shown that although acoustic features by far outperform any of the higher level verification

strategy, the fusion of all methods achieve EER relative improvements of between 28% and 48%. Ferrer et al. [37] use phone n-grams, word n-grams and some prosodic features and combine their scores with the ones from cepstral GMM, cepstral SVM and MLLR transform [38] SVM systems. [39] demonstrates N-best combination results for these systems.

Some of the papers focus on methods to enhance verification performance for systems with limited training and/or testing data. For example in [40], kernel eigenspace-based MLLR (KEMLLR) adaptation approach is proposed and compared with MAP, MLLR, and RSW. It is shown that KEMLLR adaptation outperforms for very short utterances (2-4 seconds), and results are similar with MLLR for the 8-seconds case. Xie et al. [41] apply short-time Gaussianization and kurtosis normalization on the GMMs, with a 10sec train/10sec test setup, however only achieve a slight improvement on the EER. Kwon and Narayanan propose a robust speaker identification method for very short test durations (<2 sec), which involves splitting and retraining GMM speaker models as overlapped and non-overlapped regions based on the training classification errors. [42] attempts to find a minimum duration of speech required to make a confident verification decision at a specific threshold. Vogt et al., who experimented FA modeling for short utterances, report that although error rates benefit from speaker factor dimensionality reduction, channel compensation is not an appropriate choice for training data below 20 seconds [43]. Another interesting study [44] tries to decrease the EER by finding an optimal energy threshold parameter and vector dimension which keeps a reasonable number of frames in short recordings.

The yearly Speaker Recognition Evaluation (SRE) Campaigns by NIST have become the main events which motivate speaker recognition research. It provides a joint platform for contributors to collaborate on the subject and compare their algorithms by setting a common database and evaluation setup. With the latest change in 2008, the applicants compete in 13 distinct and separate tests (categories), each of which involves one of six training conditions and one of four test conditions. The core test, which includes short2/short3 train/test conditions (formerly called 1conv/1conv) is mandatory. The training and test data contains one two-channel telephone conversation of approx-

imately five minutes total duration, or a microphone recorded conversational segment of approximately three minutes [45]. The amount of speech for each target speaker is about 2-2,5 minutes for each recording. The effect of training data is investigated in 3conv/short3 and 8conv/short3 tasks, which has 3 and 8 such recordings, respectively. Another task with increasing popularity is the 10sec/10sec task, where speaker training and testing are done over only 10 seconds of speech. Performance comparison of techniques are evaluated using the minimum decision cost function, which will be presented in Section 3.4.3.

3. THEORETICAL BACKGROUND

3.1. Definition of a Speaker Verification System

Briefly, a speaker verification system extracts parameters from the input speech signal to represent vocal characteristics of the speaker and compares these to that of the claimed speaker's. If the similarity is above a threshold, the system accepts the speaker; if not, it decides on having received an illegal access attempt.

As every pattern recognition system, speaker verification consists of three main steps: Feature extraction, modeling and testing. The block diagram of a verification task is depicted in Figure 3.1. The following sections discuss the details of each of these three blocks.

3.2. Feature Extraction

The first step in speaker verification is feature extraction, which includes representation of signals with a reduced set of mathematical entities. The properties of each observed interval of a speech signal (utterance) is represented by a feature vector in a multidimensional space. A good feature set is desired to exhibit high speaker discrimination power, high interspeaker variability and low intraspeaker variability [7].

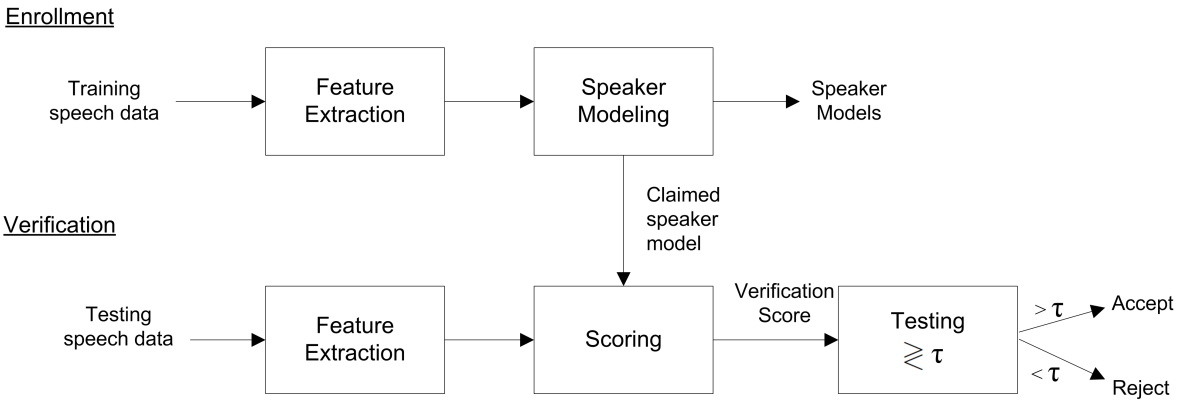

Figure 3.1. Block diagram of a speaker verification system

Several feature sets have been proposed to represent these properties, such as Linear Prediction Coefficients (LPC), log-area ratios, reflection coefficients, and MPEG-7 Normalized Audio Spectrum Envelope (NASE) features [46]. By far the most common feature set used in speaker recognition experiments is the Mel-Frequency Cepstral Coefficients (MFCC), which will also be adopted in this study.

3.2.1. MFCC Features

The reason why MFCCs are so widely used both in speech recognition and in speaker recognition is that the Mel-scale, in which the features are represented, approximates the human auditory perception mechanism with its logarithmically spaced filters. MFCCs are computed briefly as follows:

First, the speech waveform is preemphasized by a first order filter, to remove some articulatory effects and raise the energy of higher frequency regions. Then, Short Time Fourier Transform (STFT) is applied on small overlapping segments of the signal, to obtain a frequency scale representation. These frequencies are nonlinearly transformed into the Mel-scale, which has some number of (typically between 18-24) triangular filters with different center frequencies and bandwidths. Finally, Discrete Cosine Transform (DCT) is applied on the logarithmic energy of each frequency subband to obtain the cepstral coefficients.

The ΔMFC and $\Delta\Delta$MFC coefficients may additionally be computed on the first- and second-order differences of MFCCs to symbolize the spectral dynamics of speech. The energy coefficient is usually discarded to increase robustness against varying channel and recording conditions. The way preprocessing is applied on the speech signal and the number of coefficients affect speaker recognition performance to some extent. The effects of such changes can be observed in studies [3, 35, 47, 48].

3.3. Modeling

In the second step, extracted features are used to build representative models for each speaker. This corresponds to algorithmically defining the possible categories (classes) of the experiment. Two types of classifiers are preferred for this purpose: Generative and discriminative. Generative classifiers try to find a compact representation of class-dependent features, while discriminative classifiers tend to find the separating boundary between them. In the following subsections, we give details on the most popular classifier types applied to speaker verification: Gaussian Mixture Models (GMM) and Support Vector Machines (SVM).

3.3.1. Gaussian Mixture Model

A Gaussian Mixture Model (GMM) is a composition of multidimensional Gaussian distributions. The selection of Gaussians is motivated by the fact that the weights of individual components can be interpreted to reflect speaker dependent characteristics of some broader general acoustic vocal tract configuration, and that the mixture density is shown to provide a smooth approximation to the underlying distribution [3].

A Gaussian mixture consists of a weighted sum of K (D-dimensional) components, each of which is represented by a mean vector (μ) and a covariance matrix ($\mathbf{\Sigma}$):

$$\lambda_k = (\mu_k, \mathbf{\Sigma}_k) \tag{3.1}$$

$\mathbf{x}$ being the feature vector (MFCC vector in our case), the probability that it is generated by the k-th component of the mixture (the likelihood) is formulated as:

$$p(\mathbf{x}|\lambda_k) = \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}_k|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x}-\mu_k)\right\} \tag{3.2}$$

The likelihood of $\mathbf{x}$ under the mixture model is then:

$$p(\mathbf{x}|\lambda) = \sum_{k=1}^{K} \omega_k \, p(\mathbf{x}|\lambda_k) \tag{3.3}$$

where $\lambda = \{\lambda_k\}_{k=1}^{K}$ and the mixture weights ($\omega = \{\omega_k\}_{k=1}^{K}, \omega_k \geq 0$) satisfy $\sum_{k=1}^{K} \omega_k = 1$ Diagonal covariance matrices are preferred over full covariance matrices, as they are computationally more efficient (no full matrix inversions are required). They are also easier to estimate, and this makes systems which use diagonal covariance outperform the ones with full covariance [6].

Parameters of the mixture model are estimated using the Expectation-Maximization (EM) algorithm. EM is an iterative algorithm with a goal to estimate an updated model λ' based on the initial model λ, such that $p(\mathbf{x}|\lambda') \geq p(\mathbf{x}|\lambda)$.

3.3.2. UBM

For a closed-set classification problem such as speaker identification, GMM posterior probabilities of each class (probabilities of observing the i-th speaker given the feature vector, $p(\lambda^{s_i}|\mathbf{x})$) can be consistently compared with each other, to find the most likely speaker who would have created the given acoustic vector sequence. For the verification case, on the other hand, we need a reference model to which speaker models will be compared. This can either be a speaker-specific model in which we gather data of speakers whose speech are known to be similar to the target speaker (called the Cohort Model), or a single general model, which each speaker model is tested against (called the Universal Background Model).

UBM is a general speaker model which is constructed using a large amount of pooled speech data. The mixture size of UBM is determined according to the amount of speech available and the number of speakers. Besides providing a common basis for consistent evaluation of posterior probabilities, UBM also acts as a common root to all speaker GMMs. This subject is discussed in the following subsection.

3.3.3. Adaptation

To build up each speaker model, a single GMM can be trained using the speaker's training recordings. However, data sparsity may occur in this representation, especially if few enrollment data are available: The number of acoustic vectors are not adequate to "fill in" each mixture component, so that some of the weights in GMM become zero. It is common practice to derive the speaker models by adapting the parameters of the well-trained UBM, which is composed using more data. This approach is called GMM/UBM, and is shown to perform better than independently trained speaker modeling [49]. The most popular adaptation methods are Maximum-a-posteriori (MAP) Adaptation, and Maximum-Likelihood Linear Regression (MLLR) Adaptation.

3.3.3.1. MAP. Maximum-a-posteriori adaptation is based on the Bayesian estimation of model parameters, using an approach similar to the EM algorithm. The speaker models are derived by adapting the UBM using the speaker's training data and the obtained model parameters [49, 50].

The first step of the adaptation process is calculating sufficient statistics of the speaker's data. First, we match the speaker's training vectors $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_T$ to mixture components of the UBM, by computing

$$p(k|\mathbf{x}_t) = \frac{\omega_k p_k(\mathbf{x}_t)}{\sum_{k=1}^{K} \omega_k p_k(\mathbf{x}_t)} \tag{3.4}$$

We then use this probabilistic alignment to calculate the weight, mean and variance parameters,

$$\begin{aligned} n_k &= \sum_{t=1}^{T} p(k|\mathbf{x}_t) \\ E_k[\mathbf{x}] &= \frac{1}{n_k} \sum_{t=1}^{T} \mathbf{x}_t \, p(k|\mathbf{x}_t) \\ E_k[\mathbf{x}^2] &= \frac{1}{n_k} \sum_{t=1}^{T} \mathbf{x}_t^2 \, p(k|\mathbf{x}_t) \end{aligned} \tag{3.5}$$

where n_k is the number of speaker samples that correspond to UBM's k-th mixture component, and $E_k[\mathbf{x}]$ and $E_k[\mathbf{x}^2]$ represent first and second order moments, respectively. In the second step, these parameters are linearly combined with the UBM's statistics via an adaptation coefficient, α_k. This coefficient controls the balance between the weight of speaker and background models over the adapted final model parameters for each mixture component k and is defined as,

$$\alpha_k = \frac{n_k}{n_k + \tau} \ , \tag{3.6}$$

where τ is a chosen relevance factor [51]. The adapted parameters are calculated using the equations

$$\begin{aligned} \hat{\omega}_k &= \left[\alpha_k \frac{n_k}{T} + (1-\alpha_k)\omega_k\right]\gamma \\ \hat{\mu}_k &= \alpha_k E_k[\mathbf{x}] + (1-\alpha_k)\mu_k \\ \hat{\sigma}_k^2 &= \alpha_k E_k[\mathbf{x}^2] + (1-\alpha_k)(\sigma_k^2 + \mu_k^2) - \hat{\mu}_k^2 \end{aligned} \tag{3.7}$$

In speaker verification experiments, usually only the means are adapted; covariance matrices and weights are directly transferred from the UBM.

As can be seen from Equation 3.6, if a mixture component has a low probabilistic count of adaptation data, then $\alpha_k \to 0$, which increases the emphasis of the well-trained UBM model parameters over the final values. For mixture components of high probabilistic counts, on the other hand, $\alpha \to 1$, and the system trusts more on the new observation parameters. The relevance factor τ controls how much new data should be observed before the new parameters take over the old ones. Selection of a larger τ leads to models identical to the UBM whereas a smaller τ discards the effect of adaptation [49].

The main limitation of MAP adaptation is that the estimation accuracy depends on the amount of adaptation data. Besides, MAP requires an accurate initial prior

guess, which is often difficult to obtain. It is also reported that when a large number of free parameters need to be adapted, the process can be very slow [52].

3.3.3.2. MLLR. MLLR constructs speaker models by updating the UBM means with an appropriate linear transformation [53, 54]. The parameters of this transformation are calculated by linear regression. If $\mathbf{A}$ is the regression matrix and $\mathbf{b}$ is an additive bias vector, the adapted mean vector of the k-th mixture component becomes,

$$\hat{\mu}_k = \mathbf{A}\mu_k + \mathbf{b} \tag{3.8}$$

Higher order statistics are not adapted, as the main differences between speakers are assumed to be characterized by the means.

MLLR is a very effective method for rapid adaptation, since the transformation parameters can be estimated from a relatively small amount of data. When the amount of adaptation data is limited, it offers better overall performance. When the amount of adaptation data increases, MAP becomes more accurate because we can modify all the model parameters with MAP training [52]. This characteristic can also be used to alleviate the problem with MAP: First MAP is used to adapt model parameters, then MLLR is applied to transform (smooth) the adapted parameters [55].

3.3.4. Support Vector Machines

A Support Vector Machine (SVM) is a binary linear classifier, which aims to find a separating hyperplane that maximizes the margin between the nearest samples of two classes. Geometrically this amounts to locating the separating hyperplane in a perpendicular direction, midway along the shortest line separating the convex hulls of these classes. The samples inside and on the borders of this margin are called the support vectors.

Consider the binary classification problem depicted in Figure 3.2. Assume that

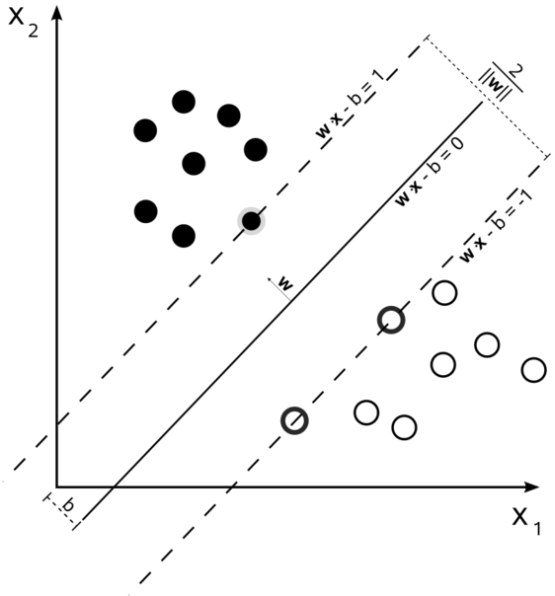

Figure 3.2. An example SVM setup

the decision hyperplane is expressed by the formula,

$$f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{w} + b = 0 \tag{3.9}$$

where $\mathbf{w}$ is its normal; and that all training data satisfy,

$$(\mathbf{x}_i \cdot \mathbf{w} + b)\, y_i \geq 1 \quad \forall i = 1, \ldots, N \tag{3.10}$$

where y_i's denote the class labels, i.e., $y_i \in \{-1, 1\}$.

The data points $\mathbf{x}_i$ which yield equality in the formula above are said to be on the marginal hyperplanes. Maximizing the distance between marginal hyperplanes is equivalent to minimizing the objective function

$$E = ||\mathbf{w}||^2 \ . \tag{3.11}$$

This constrained optimization problem is solved by introducing Lagrange multipliers

$\alpha_i \geq 0$ in the Lagrangian

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

where $L(\mathbf{w}, b, \alpha)$ is simultaneously minimized with respect to $\mathbf{w}$ and b and maximized with respect to α_i. Solving this optimization problem leads to the decision boundary:

$$f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{w} + b = \sum_{i=1}^{N} y_i \, \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b = 0 \tag{3.13}$$

If the data are not linearly separable, the objective function is reformulated by introducing slack variables ξ_i, which allow some samples to violate the margin constraints. The problem which Equations 3.10 and 3.11 denote then becomes:

$$\begin{aligned} E &= \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{N} L(\xi_i) \\ \text{subject to } &(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i \quad \forall i = 1, \ldots, N \end{aligned} \tag{3.14}$$

Here, $L(\cdot)$ is a distance metric (generally the L1- or L2-norm) and C is a user-defined penalty (cost) parameter to penalize violations of the safety margin. A larger C leads to a narrower margin, thus fewer SVs. It is also possible to extend this model by introducing class-dependent costs, especially if it is more severe to misclassify one class against the other.

SVMs can as well be adapted to create nonlinear boundaries between classes, by the help of the "kernel trick": Linearly non-separable data points in the input space are mapped by a nonlinear function $\phi(\mathbf{x})$ to a higher (possibly infinite) dimensional feature space, in which they are linearly separable. The decision boundary (Equation 3.13) is then expressed as:

$$f(\mathbf{x}) = \phi(\mathbf{x}) \cdot \mathbf{w} + b = \sum_{i=1}^{N} y_i \, \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b = 0, \tag{3.15}$$

where the kernel $K(\cdot,\cdot)$ supports the Mercer condition and can be written as an inner product of the transformed data points:

$$K(\mathbf{x},\mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y}) \tag{3.16}$$

3.3.5. GMM Supervector Approach

There are several ways to create input features for SVMs. Older approaches include training SVMs directly on the acoustic vectors [13], or using GMM/UBM classifier output scores (likelihood ratios) as sample vectors [14]. To use a whole utterance as a feature vector, we need some kind of a transformation which processes the vector sequence to output a single vector of constant dimension, so that different utterances can be represented by different samples in the SVM space. To achieve this, the method by Campbell et al. [20] uses Generalized Linear Discriminant Sequence (GLDS) kernel which utilizes polynomial expansions.

Lately, the most popular feature extraction method for speaker verification in an SVM setup is called the GMM supervector approach, proposed by Campbell et al. [16]. The procedure is as follows: Consider again the UBM presented in Section 3.3.2. Acoustic features are first extracted from a given utterance. GMM training is performed by MAP adaptation of the means of the UBM model. The GMM supervector is then constructed by appending the adapted means of each mixture component in a single high dimensional vector. For instance, for a UBM model of K mixtures and D dimensions, the GMM supervector is of size $KD \times 1$. This process is illustrated in Figure 3.3.

3.3.6. SVM Design Issues

Two more choices need to be made before constructing the SVM classifier for the speaker verification experiment. The first one is selecting an appropriate kernel, and the second, deciding on the type of the classifier. In some cases the samples of one

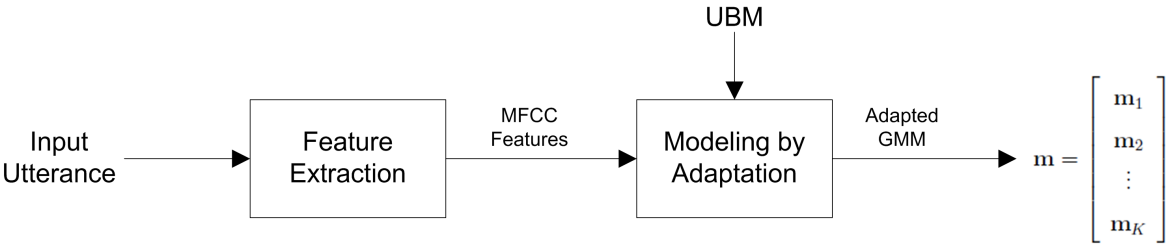

Figure 3.3. Determination of GMM supervector

class may be smaller in number than the other one's. In such a case, data balancing could also be applied before training the SVM. The following subsections expand on these issues.

3.3.6.1. Kernel Type. Comparing Equations 3.13 and 3.15, we see that the inner product of two vectors,

$$K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x} \cdot \mathbf{y} \tag{3.17}$$

is in fact the simplest kernel type, which is called the linear kernel. Two other kernels extensively used in pattern recognition applications are the Radial Basis Function (RBF) kernel (also called the Gaussian kernel [56], or the non linear kernel [23, 25]) which is defined by

$$K(\mathbf{x}, \mathbf{y}) = \exp\left\{-\frac{||\mathbf{x} - \mathbf{y}||^2}{2\sigma^2}\right\} \tag{3.18}$$

and the polynomial kernel which is formulated as

$$K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x} \cdot \mathbf{y} + 1\right)^p . \tag{3.19}$$

The σ and p in these equations denote the influence area parameter of the basis function and degree of the polynomial, respectively. Another kernel worth mentioning is the GMM supervector linear kernel, introduced by Campbell et al. [16, 22]. This type of kernel calculates a distance between two supervectors $\mathbf{m}^a$ and $\mathbf{m}^b$, derived from the

corresponding GMMs λ^a and λ^b, based on the Kullback-Leibler (KL) divergence:

$$KL(\lambda^a||\lambda^b) = \int_{R^n} \lambda^a(\mathbf{x}) \log\left(\frac{\lambda^a(\mathbf{x})}{\lambda^b(\mathbf{x})}\right) d\mathbf{x} \tag{3.20}$$

Since the KL divergence does not satisfy Mercer condition, an approximation is considered. An upper bound to Equation 3.20 is

$$KL(\lambda^a||\lambda^b) \leq \sum_{k=1}^{K} \omega_k \; KL\Big(\mathcal{N}(\cdot, \mathbf{m}_k^a, \mathbf{\Sigma}_k)||\mathcal{N}(\cdot, \mathbf{m}_k^b, \mathbf{\Sigma}_k)\Big) \tag{3.21}$$

If we assume Σ_ks to be diagonal, the approximation in Equation 3.21 simplifies to

$$d(\mathbf{m}_k^a, \mathbf{m}_k^b) = \frac{1}{2} \sum_{k=1}^{K} \sum_{d=1}^{D} \omega_k \left(\frac{\mathbf{m}_{kd}^a - \mathbf{m}_{kd}^b}{\sigma_{kd}}\right)^2 \tag{3.22}$$

Considering Equations 3.20 and 3.21, we conclude that if the distance between the supervectors is small, the corresponding divergence is small. Using this symmetric distance notation, the kernel function is expressed as:

$$\begin{aligned} K(\mathbf{m}_k^a, \mathbf{m}_k^b) &= \sum_{k=1}^{K} \omega_k (\mathbf{m}_k^a)^T \mathbf{\Sigma}_k^{-1} \mathbf{m}_k^b \\ &= \sum_{k=1}^{K} \left(\sqrt{\omega_k}\mathbf{\Sigma}_k^{-\frac{1}{2}}\mathbf{m}_k^a\right)^T \left(\sqrt{\omega_k}\mathbf{\Sigma}_k^{-\frac{1}{2}}\mathbf{m}_k^b\right) \end{aligned} \tag{3.23}$$

3.3.6.2. Classifier Type. There are two basic strategies for generating multi-class SVMs. In One-Against-One (OAO) type SVMs, pairwise binary classifiers are constructed for each couple of C classes. Therefore, the number of classifiers is $C(C-1)/2$. Usually, classification is done according to the majority voting principle [57].

OAO type SVMs are the typical choice for speaker identification applications [58]. For speaker verification, the other strategy, namely the One-Against-All (OAA) SVMs are used. In OAA SVMs, a single classifier is constructed for each class, which makes up a total of C classifiers. Each of these setups use samples of a target class as

the positive examples against a collection of negative examples. For general pattern recognition applications, this collection includes all samples from the other $(C-1)$ classes (hence also called the One-Against-Rest setup). In speaker verification, the preference is to use a set of common background samples instead of other classes, which can be named as the One-Against-Background setup. We adopt the latter type in our experiments.

3.3.6.3. SVMs with Imbalanced Data. One of the major drawbacks of SVM classifiers is that their success is limited when the number of examples in one class is very small than the other. This issue is called "class imbalance" and the positive and negative examples are denoted as minority and majority classes, respectively. Application areas such as medical diagnosis and credit card fraud detection have highly imbalanced datasets with a very small number of positive instances which are hard, but important to classify correctly [59]. This is also the case for One-Against-Background SVMs applied to speaker verification, where the number of background samples heavily outnumber the speaker's training samples. Several methods have been proposed to cope with class imbalance. A comprehensive background on reasons and possible solutions can be found in [59] and [60]. We mention two of these techniques here.

In an imbalanced data case, to reduce the overwhelming errors of misclassifying the majority class, the optimal hyperplane will inevitably be skewed to the minority; so that on the extreme case, SVM learns to classify everything as negative [61]. To prevent this, the cost associated with the positive samples could be increased [62]. Yuan et al. report that for linearly separable data, tuning the costs has little effect, while for the linearly non-separable case it changes the position of the separating hyperplane considerably [61].

Another group of approaches tries to solve the class imbalance problem by oversampling the minority class, or undersampling the majority class [60]. Akbani et al. show that although undersampling the majority class improves SVM performance, it should not be preferred due to the inherent loss of valuable information in this pro-

cess [59]. On the other hand, oversampling is also believed to be an undesired way of duplicating data, as it introduces an unnatural bias in favor of the minority class [60].

An intelligent method to oversample the positive class is the Synthetic Minority Oversampling Technique (SMOTE), proposed by Chawla et al. In this technique, nearest neighbors of each positive instance are identified and the new positive examples are generated randomly in between the sample and its neighbors [63]. We investigate the effect of SMOTE in Section 6.5.

3.4. Testing and Evaluation

To assess and compare the performance of speaker verification systems, we need some output scores and evaluation metrics built upon these. The first two subsections present how the output scores are obtained in GMM and SVM setups, respectively. The subsection that follows explains how these scores are used to compare verification systems.

3.4.1. GMM/UBM Scoring

A speaker verification experiment with a GMM/UBM setup can be viewed as a hypothesis testing problem, where we have to decide whether a given utterance $\mathbf{X}$ is spoken by the speaker S, whom it claims to be. Let us define here the two hypotheses as:

$\mathcal{H}_1$: $\mathbf{X}$ belongs to speaker S
$\mathcal{H}_0$: $\mathbf{X}$ does not belong to speaker S / $\mathbf{X}$ is an impostor attempt

Here, $\mathcal{H}_1$ is represented by the claimed speaker model λ^S and $\mathcal{H}_0$ is represented by the universal background model λ^{UBM}. This idea is illustrated in Figure 3.4 with a single Gaussian model for a one dimensional distribution.

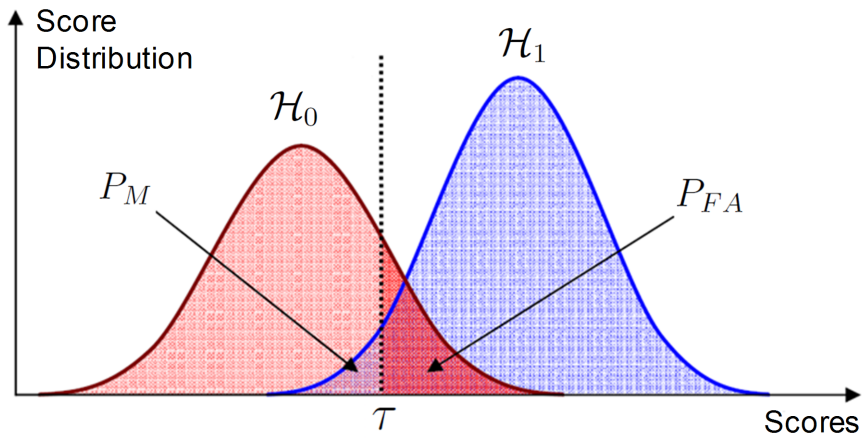

Figure 3.4. Hypothesis testing for verification

The optimal decision in this setup is defined by the likelihood ratio test, formulated as:

$$\Lambda(\mathbf{X}) = \frac{p(\mathbf{X}|\lambda^S)}{p(\mathbf{X}|\lambda^{UBM})} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \tau \tag{3.24}$$

For our GMM case, this function is expressed as a product of sequence vector probabilities:

$$p(\mathbf{X}|\lambda) = \prod_t p(\mathbf{x}_t|\lambda) \tag{3.25}$$

In practice, to prevent numerical underflow, log-sum is used instead of product operation, which converts Equations 3.24 and 3.25 respectively into,

$$\log \Lambda(\mathbf{X}) = \log p(\mathbf{X}|\lambda^S) - \log p(\mathbf{X}|\lambda^{UBM}) \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \tau' \text{ , and,} \tag{3.26}$$

$$\log p(\mathbf{X}|\lambda) = \sum_t \log p(\mathbf{x}_t|\lambda) \ . \tag{3.27}$$

The likelihood score, dependent on the decision threshold τ (or τ'), is used as the decision metric on whether to accept the utterance as a successful client attempt.

3.4.2. SVM Scoring

The usual decision metric of an SVM classifier is the decision function value $f(\mathbf{x})$ of Equation 3.15. The sign of $f(\mathbf{x})$ determines the class of which $\mathbf{x}$ is assigned to, and the value itself shows how close the test sample is to the decision boundary. For One-Against-All type SVMs, a method to convert these decision function values into real probabilities has been proposed by Platt [64] and later improved by Lin and Weng [65]. The conversion is formulated as

$$p(y = 1|f^*(\mathbf{x})) = \frac{1}{1 + \exp(Af^*(\mathbf{x}) + B)} \tag{3.28}$$

where A and B are estimated by minimizing the negative log-likelihood function using the training data and their deterministic maximal margin classifier decision values, $f^*(\mathbf{x})$.

3.4.3. Evaluation Metrics

The final verification decision on whether an utterance belongs to the claimed identity is given by comparing the output score (whether the likelihood ratio as in section 3.4.1 or the function value or probability in section 3.4.2) to a threshold τ. Two types of errors may occur in this decision: Misses (false rejections) and false alarms (false acceptances). A miss happens when a valid identity claim is rejected, whereas a false alarm occurs when an impostor attempt is accepted.

The tradeoff between false alarm and miss rates, shown by P_{FA} and P_M respectively, depend on the threshold (see Figure 3.4). By changing this value, it is possible to move over different operating points ((P_{FA}, P_M) pairs) of the system. The plot which shows all possible operating points of a system is called the Receiver Operating Characteristic (ROC) curve (Figure 3.5).

The selection of the operating point depends on security requirements of the verification task. A highly secure system would prompt the user to provide another

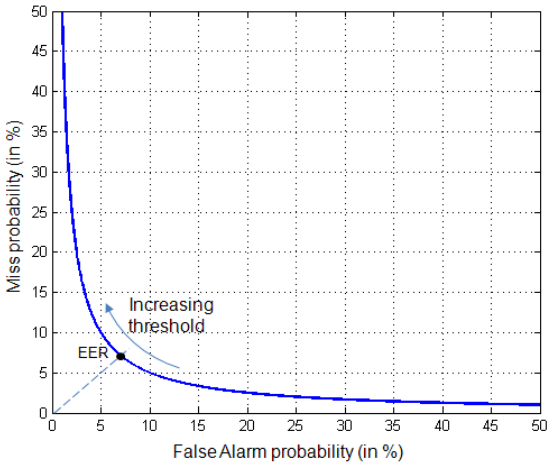

Figure 3.5. A typical ROC curve

test sample before making a decision, to avoid false alarms. On the other hand, some applications may set the decision threshold too low especially if misses cannot be tolerated. To compare different verification systems, a traditional measure is the Equal Error Rate (EER), the operating point where P_{FA} equals P_M. In speech applications it is also found helpful to use a variant of the ROC curves, plotted on a normal deviate scale. This type of plots is called Detection Error Tradeoff (DET) curve. DET curves produce plots close to linear and thus help better distinguish performances of different systems [66]. Figure 3.6 depicts an example of a DET curve.

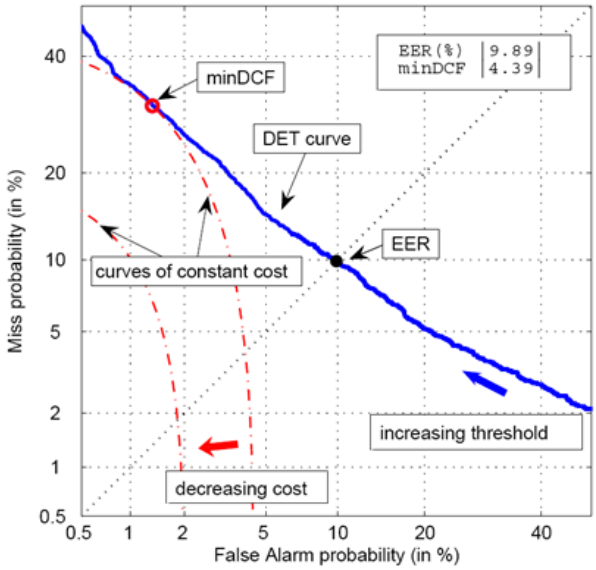

Figure 3.6. A typical DET curve [67]

Another measure to compare systems is the minimum detection cost function

(minDCF) value, visible in Figure 3.6. The detection cost function (C_{Det}), the official performance measure of NIST Speaker Recognition Evaluation Campaigns, is a weighted sum of miss and false alarm probabilities:

$$C_{Det} = C_M \; P_M \; P_{target} + C_{FA} \; P_{FA} \; (1 - P_{target}) \tag{3.29}$$

Here, C_M and C_{FA} are relative costs attributed to miss and false alarms, and P_{target} is the a priori probability of the specified target speaker. In NIST SRE tasks, these parameters are set as,

C_M	C_{FA}	P_{target}
10	1	0.01

The C_{Det} value is further normalized by the best cost that could be obtained without processing the input data (either setting $P_M = 1$ or $P_{FA} = 1$) to get a more intuitive score at that specific operating point ($C_{NormDet}$). The minDCF value is defined as the minimum of all possible $C_{NormDet}$ scores [45].

3.4.4. Normalization Techniques

For a speaker verification system which uses clean speech with a single session and fixed recording conditions, the scores obtained from decision making process is consistent. However, for a system having multiple recording environments and changing constraints, which is most likely the case for a realistic application, these scores are observed to have high variability between trials and thus are not very reliable.

Various reasons account for score variability. One of these reasons is the acoustic mismatch between training and test data. Acoustic mismatch can occur as a result of changes in the speaker's voice: The speaker may be in a different emotional state (happy, sad, angry, etc.) so that the acoustic properties of the speech signal change. Health status is another important issue; vocal tract characteristics are altered when the person is sick. Aging effects are important too, especially in children and teenagers.

Apart from voice characteristics, changes in recording environment also create acoustic mismatch: Noise is an important obstacle against making correct decisions, and variations in recording device and transmission environment are considered to be one of the greatest problems of speaker verification. Finally, the phonetic content, the duration, and the quality of the speaker models are among the other reasons for score variability. All these reasons may change between different speakers of the same session (called speaker variability), or between recording sessions of the same speaker (called session variability).

Several score normalization techniques are proposed to compensate for mismatches by reducing the variance of overall score distribution. These techniques intend to calculate a mean (μ) and a standard deviation parameter (σ) over a pseudo-impostor score distribution, and normalize the original scores by,

$$\widetilde{\Lambda}(\mathbf{X}) = \frac{\Lambda(\mathbf{X}) - \mu}{\sigma} \tag{3.30}$$

where μ and σ can denote parameters for different speakers or test utterances. A comprehensive summary of these methods can be found in [6]. Two mostly used methods will be presented here: Z-Norm and T-Norm.

3.4.4.1. Z-Norm. A primitive score normalization technique, derived from the study by Li and Porter [68] is the zero normalization (Z-Norm). Each speaker model (λ^{s_i}) is tested against a set of held-out pseudo-impostor utterances, $\{\mathbf{X}_I\}$. A mean (μ^{s_i}) and standard deviation parameter (σ^{s_i}) are estimated from the distribution of these decision scores, $\{p(\mathbf{X}_I|\lambda^{s_i})\}$. These parameters are then used to normalize the original scores, as formulated in Equation 3.31. Determination of parameters can be performed offline before the actual testing phase.

$$\widetilde{\Lambda}^{s_i}(\mathbf{X}) = \frac{\Lambda^{s_i}(\mathbf{X}) - \mu^{s_i}}{\sigma^{s_i}} \tag{3.31}$$

3.4.4.2. T-Norm. One of the most widely used normalization schemes is test normalization (T-Norm), proposed by Auckenthaler et al. [27]. Instead of the pseudo-impostor test data in Z-Norm, T-Norm uses a set of pseudo-impostor models ($\{\lambda^I\}$), which the original test data will be tested against. Therefore, normalization parameters are computed for each test utterance over a collection of pseudo-impostor scores, $\{p(\mathbf{X}|\lambda^I)\}$, and normalization is applied as,

$$\widetilde{\Lambda}(\mathbf{X}) = \frac{\Lambda(\mathbf{X}) - \mu^I}{\sigma^I} \tag{3.32}$$

Since the same test utterance is used during both the testing and parameter estimation, T-Norm also avoids a possible acoustic mismatch of test data, observed in Z-Norm. The drawback of T-Norm is that normalization parameters cannot be computed beforehand [6].

3.4.5. Nuisance Attribute Projection

Nuisance Attribute Projection (NAP) is one of the popular methods to mitigate the problem of channel variability, which is the acoustic mismatch when a speaker is enrolled on one type of channel and is tested on one another. The basic idea of NAP is to remove dimensions from the SVM space that represent the channel effects ("nuisance attributes"), thus allowing only speaker variability [26]. This is achieved by projecting out a subspace from the original space using an appropriate projection matrix P.

The details of this method are as follows [24]: Let $\mathbf{m}_{h_j}^{s_i}$ denote the supervector (or some other expansion form) of the i^{th} speaker (s_i) at the j^{th} session (h_j). Also assume that we have S speakers, and H sessions. First, a session-averaged supervector ($\bar{\mathbf{m}}^{s_i}$) is calculated for each speaker. This value is then removed from all the corresponding examples to obtain their mean-shifted versions

$$\widetilde{\mathbf{m}}_{h_j}^{s_i} = \mathbf{m}_{h_j}^{s_i} - \bar{\mathbf{m}}^{s_i} \tag{3.33}$$

The matrix

$$\mathbf{M} = [\widetilde{\mathbf{m}}_{h_1}^{s_1} \cdots \widetilde{\mathbf{m}}_{h_H}^{s_1} \cdots \widetilde{\mathbf{m}}_{h_1}^{s_S} \cdots \widetilde{\mathbf{m}}_{h_H}^{s_S}] \tag{3.34}$$

represents all the intersession variations from the average speaker positions and has a size of $E \times N$. We assume that a subspace of dimension $K < N$, where the variations are the greatest, represent the space spanned by channel characteristics. We therefore calculate the K eigenvectors with the highest eigenvalues of the covariance matrix $\mathbf{C} = \mathbf{M}\mathbf{M}^T$. The resulting eigenvectors form a base to the reduced channel subspace $\mathbf{R}$ of size $E \times K$. The projection operation for any supervector $\mathbf{m}_x$ is then defined as,

$$P(\mathbf{m}_x) = (\mathbf{I} - \mathbf{R}\mathbf{R}^T)\mathbf{m}_x \tag{3.35}$$

Finally, the modified supervectors to be used in the SVM framework are formulated as,

$$\hat{\mathbf{m}}_x = \mathbf{m}_x - \mathbf{R}(\mathbf{R}^T\mathbf{m}_x) \tag{3.36}$$

NAP has been a favorite technique to remove channel effects in NIST evaluations, along with its counterpart, Factor Analysis. For the experiments which use a single training example, it is not possible to calculate the channel variation matrix over the training data. In such case, NAP subspace is computed on a held-out development set.

NAP is successfully applied to the SVM setup with linear kernels. However, for nonlinear kernels, solving the eigenvalue problem in the high dimensional space brings with numerical difficulties in matrix calculations. To overcome this problem, kernel PCA is used instead of PCA [30].

3.4.6. Fusion Techniques

The idea behind fusion is to combine inputs or outputs of two different decision systems to finally arrive at a more powerful classification (or verification) decision. For speaker verification, fusion can be done either at the feature level, or at the score level.

Feature fusion is the method where a combination of different features are fed into the same classifier. For instance, MFCC and LPCC features may be combined in a GMM/UBM network. Another setup would be the fusion of MAP- and MLLR-adapted supervectors in an SVM setting. This approach is applied in this study and will be presented in Section 5.2.

Over the score fusion techniques, the most widely used ones are linear fusion [33], linear logistic regression [35, 69], and neural networks with single- and multi-layer perceptrons [28, 36]. One important issue to take into account is that the fused scores must share a similar range, i.e., they must be normalized. Speaker verification systems usually apply T-Norm for this purpose.

Linear fusion involves calculating a weighted sum of the scores of different decision systems, so that the fusion score becomes,

$$s_f = \beta_1 s_1 + \beta_2 s_2 + \ldots + \beta_N s_N \tag{3.37}$$

Many of the studies which use linear fusion prefer choosing equal weights for normalized GMM and SVM scores [15, 25, 28].

For the linear logistic regression fusion, the weights $\beta_1, \ldots, \beta_N$ are determined by minimizing the logistic function [70]:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{3.38}$$

where $z = \beta_0 + \beta_1 s_1 + \beta_2 s_2 + \ldots + \beta_N s_N$ and β_0 is an offset factor.

4. BASELINE EXPERIMENTS

4.1. Dataset

The CSLU Speaker Recognition Corpus (version 1.1), collected by OGI and published by LDC, has been used for the evaluation. The dataset consists of telephone speech of 91 speakers, recorded in 12 sessions for over a two-year period. Utterances contain answers to short questions, repetition of words, numbers and phrases, and a short duration of spontaneous speech. For each session there exists about 100 utterances for a total duration of around 4 minutes. 44 male and 47 female speakers have participated in the collection [71].

4.2. Partitioning of Data, Training and Testing Setups

We selected 90 speakers (44 male, 46 female) and divided into three groups: UBM, user, and background sets. The UBM subset contains utterances of 20 speakers. The background subset is constituted of another group of 20 speakers, which provide samples of the SVM reference class (background class) and the impostors (cheaters who would like to get illegal access to the system by claiming a false identity). These two subsets contain equal number of male and female speakers. Finally, the last 50 speakers are labeled as the actual (registered) users for this experiment.

The UBM model is trained with all 12 sessions of its 20 speakers. Session numbers are labeled 1 through c. In the user and background subsets, the first 6 sessions (1-6) are reserved for training purposes, while the other 6 (7-c) are kept for testing.

To investigate the effects of the amount of data on speaker verification performance, several training and testing durations, which will be called "protocols" from now on, are experimented throughout the thesis. Both for GMM/UBM and SVM setups, we have three main protocols: 4min/4min, 1min/1min and 10sec/10sec. For the 4min/4min case, the GMM speaker models are trained using utterances from a single

session, while for 1min/1min and 10sec/10sec cases, training is done over a single utterance (or a collection of shorter utterances) which has the indicated total duration. As for the SVM setup, each such session (or collection) corresponds to a training sample supervector.

Testing is likewise done over these three different durations using all data in the last six sessions (7-c). The testing scenario is defined as follows: The test set consists of samples from both registered users and impostors. In the first four sessions (7-a), user-subset tests (access requests to the system) have a true claimed speaker id (valid/legal access). For the last two sessions (b-c) of this group, claimed ids are randomly assigned, which symbolizes invalid/illegal attempts. In addition, the impostor part contains speech from 20 speakers not enrolled to the system. To equalize the number of total valid and invalid attempts, 5 test sessions (out of 6) are used for the impostor testing. Table 4.1 summarizes the distribution of training and test data over sessions.

Table 4.1. Distribution training/test subsets over sessions (v:valid, i:invalid attempt)

Subset	Session Label											
	1	2	3	4	5	6	7	8	9	a	b	c
UBM Train	x	x	x	x	x	x	x	x	x	x	x	x
Background Train	x	x	x	x	x	x						
Impostor Test							x	x	x	x	x	
User Train	x	x	x	x	x	x						
User Test							v	v	v	v	i	i

4.3. Platform and Tools

The GMM/UBM setup is implemented using The BioSecure Reference System BECARS/HTK. This system includes three open-source software packages: HTK [72] for feature extraction, UNIANAL [73] for pitch, energy determination and voice activity detection, and BECARS [51] for GMM modeling and scoring [74]. The System is originally organized to work with the BioSecure Reference Database, which is the

speech part of the BANCA Database [75]. Codes and scripts are modified in order to comply with the CSLU Dataset.

LIBSVM library is selected for SVM implementation, as it enables the usage of a user-defined kernel and can provide probabilistic score outputs [76].

4.4. Feature Extraction for GMM/UBM

We use MFCCs as acoustic features to represent speaker characteristics. Speech signal is segmented with 20ms Hamming window by 10ms frame shifts. 16 MFC coefficients are extracted from each frame and the energy parameter is calculated. Δ-MFCCs and Δ-energy are appended to this vector. To determine the frames corresponding to speech portions of the signal, voice activity detection is applied by bi-Gaussian modeling of the energy component. Next, the cepstral vectors are normalized so that they have zero mean and unit variance. Finally, the energy coefficient of the vectors is discarded and the frames corresponding to silence are deleted. In the end, we have a 33-dimensional vector for each selected frame.

4.5. GMM/UBM Baseline

To investigate speaker verification performance of GMM/UBM classifiers with respect to changing model sizes under different data durations, we constructed UBM models with three different number of mixture components: GMM16 (with 16 components), GMM64 (with 64 components) and GMM256 (with 256 components). As stated in Section 4.1, 12 sessions of 20 speakers are used for this composition, which amounts to about 960 minutes of speech. GMM speaker models are adapted from these UBMs, using the entire set of utterances (for 4min/4min), or a collection of utterances (for 1min/1min and 10sec/10sec) from the first session. The MAP adaptation relevance factor is chosen as 14 (Section 5.1 presents verification performance results due to changes in this value and the adaptation method). We follow the common usage of only adapting the means and leaving the covariance and weights intact. Table 4.2 presents the EER and minDCF values for three different protocols with respect to

GMM sizes (number of components).

Table 4.2. EER and minDCF values for the baseline GMM/UBM experiments

	GMM/UBM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
4min/4min	15.95	0.0895	14.28	0.0770	12.50	0.0720
1min/1min	20.37	0.0984	21.49	0.0933	31.61	0.0862
10sec/10sec	31.21	0.0993	43.26	0.0988	48.35	0.0986

It can be seen from Table 4.2 that the EER and minDCF values decrease as we increase the training duration, which implies that speaker characteristics can be better modeled by using larger amounts of data. It also suggests that there exists an optimal mixture size for a given protocol. For 4min/4min, GMM256 best models the acoustic variability, whereas in 1min/1min and 10sec/10sec, the amount of data is not sufficient to be represented by such higher-order mixtures. It is surprising to see that, independent of the fluctuations in EER, there is a stable decrease in minDCF values with increasing model size.

Figure 4.1 shows the DET curves of 4min/4min and 10sec/10sec protocols for three model sizes. Superior performance of 4min/4min with respect to 10sec/10sec can be observed. Another notable point is the dominance of GMM16 on modeling the 10sec/10sec case.

4.6. SVM Baseline

We repeat the experiments using an equivalent setup on the SVM classifier with a supervector approach. The MAP-adapted GMM models of the GMM/UBM implementation are used to create the supervectors. Since each mean vector is 33-dimensional, we have 33×16, 33×64 and 33×256 dimensional vectors for our three main protocols.

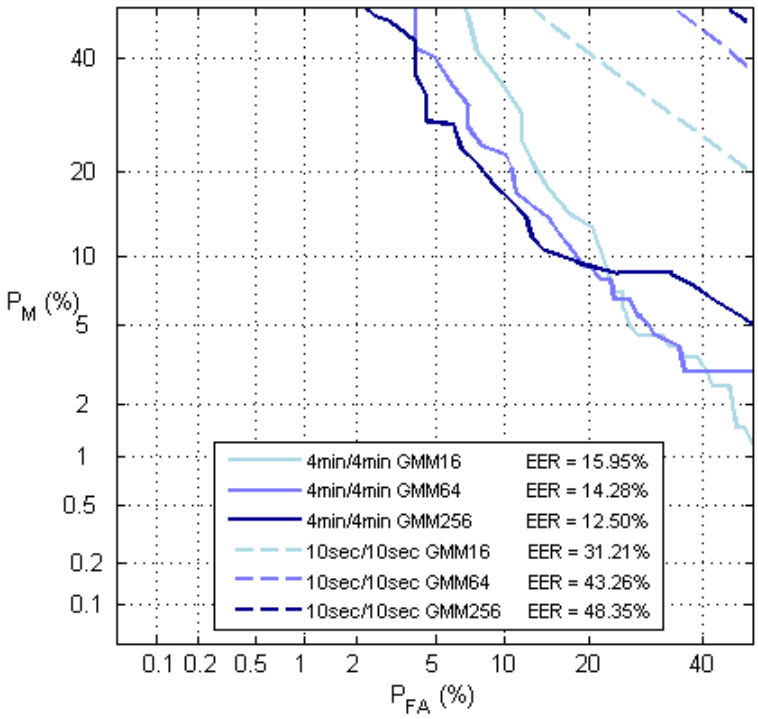

Figure 4.1. GMM/UBM baseline experiments DET curve

For each speaker, we construct a one-against-background SVM classifier, where the speaker class (positive class) is represented by a single supervector, against a collection of background (negative class) supervectors. Since we have 20 speakers in our background subset and use 6 training sessions for each, the number of background supervectors is 20×6×(number of collections). We use the inner product linear kernel $K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$ and set the cost (penalty) value as $C = 1$ in the baseline evaluation. The decision function value $f(\mathbf{x})$ in Section 3.15 is employed as the verification score. Table 4.3 shows performance metrics of the baseline SVM experiments for all nine protocol and model size combinations.

The results verify the superiority of the overall classification performance of SVM over that of the GMM/UBM. Again, as the training/testing duration is increased, error rates and minDCF values tend to decrease. Unlike the GMM/UBM case, for 4min/4min and 1min/1min protocols we also observe significant performance improvement with increasing model sizes, since a better separating hyperplane can be found

Table 4.3. EER and minDCF values for the baseline SVM experiments

	SVM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
4min/4min	15.53	0.0625	8.04	0.0435	6.00	0.0190
1min/1min	28.35	0.0837	19.72	0.0711	18.26	0.0590
10sec/10sec	36.92	0.0957	36.30	0.0930	37.60	0.0928

in those higher-dimensional spaces. Similar behavior cannot be observed with the 10sec/10sec protocol, though. The reason of this might be twofold: First, one single supervector adapted from only 10 seconds of speech may not be able to exhibit an adequate representation of speaker's vocal characteristics. Second, with the 10sec/10sec protocol, the number of background samples becomes so large that they may overdominate the feature space, even when the dimensionality is increased. Figure 4.2 depicts SVM baseline verification performance of the 4min/4min and 10sec/10sec protocols.

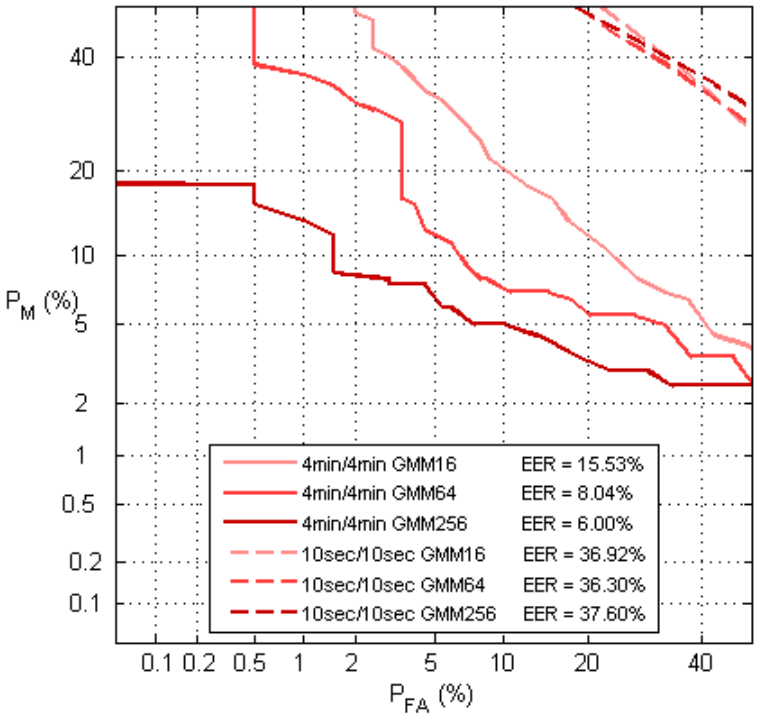

Figure 4.2. SVM baseline experiments DET curve

4.7. GMM/UBM - SVM Fusion

We applied score-level fusion on the GMM/UBM and SVM baseline tests using two strategies: Linear fusion combines the outputs of two systems with an equal weight ($\beta = 0.5$) and logistic linear fusion selects the best β parameters by optimizing over the test set.

Table 4.4. Linear Fusion of Baseline Experiments

	Linear Fusion					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
4min/4min	16.18	0.0895	14.06	0.0770	12.50	0.0715
1min/1min	20.01	0.0976	21.36	0.0933	31.26	0.0863
10sec/10sec	31.01	0.0992	42.59	0.0987	47.54	0.0986

Table 4.5. Logistic Linear Fusion of Baseline Experiments

	Logistic Linear Fusion					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
4min/4min	13.36	0.0605	8.76	0.0430	5.50	0.0195
1min/1min	21.48	0.0831	19.47	0.0708	18.71	0.0611
10sec/10sec	32.42	0.0962	36.35	0.0933	36.94	0.0935

Tables 4.4 and 4.5 contain the results of fusion strategies. Comparing these results with the ones of the individual experiments, we can deduct that logistic fusion favors the SVM classifier's outputs, and that linear fusion is better when GMM/UBM error rates are lower than SVM's. Nevertheless, the obtained results are mixed and unconvincing. We have also tried applying T-Norm on both systems before fusion, but (probably because of the lack of impostor training data), the effect of normalization worsened the performance of both classifiers. Thus, we do not present the numerical details of this experiment.

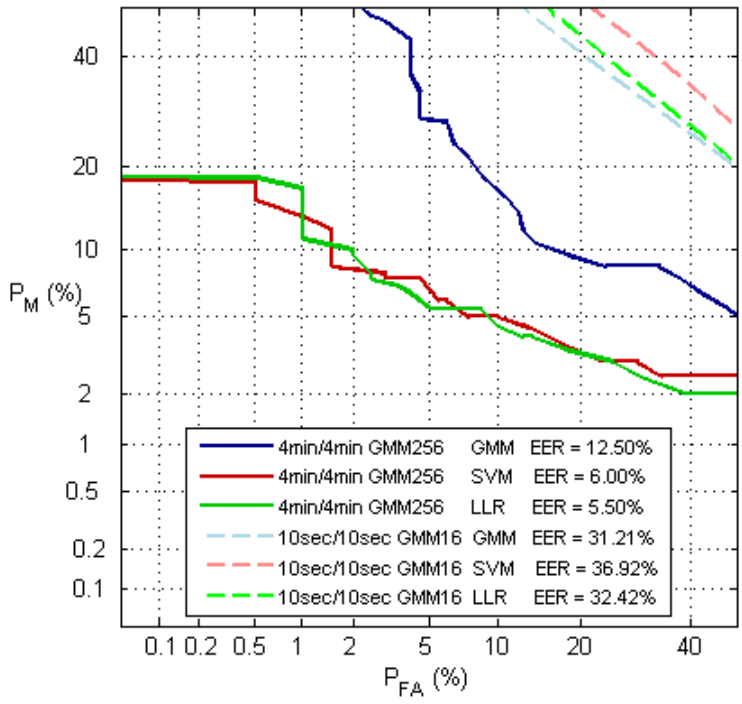

Figure 4.3. LLR Fusion DET curves for 4min/4min GMM256 and 10sec/10sec GMM16

Figure 4.3 illustrates the behavior of logistic linear fusion in comparison with corresponding GMM and SVM performances, for two sample protocols. It is possible to see that, in accordance with the EER results, LLR curve follows SVM closely in 4min/4min, whereas in 10sec/10sec it positions itself near the GMM/UBM, although not being able to surpass it.

4.8. Experiments with Constant Test Duration

Although they reflect the influence of data amount on verification accuracy, results given for different protocols in Tables 4.2 and 4.3 are not directly comparable, as they are not tested over the same utterance durations. To see how the GMM/UBM and SVM systems behave on common test conditions, we set the test data duration to 10 seconds and repeat the experiments. Tables 4.6 and 4.7 show recomputed EER and minDCF values.

Comparing the two tables, we conclude that for a given short-duration test file,

Table 4.6. Baseline GMM/UBM experiments with constant test duration

	GMM/UBM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
4min/10sec	19.49	0.0995	18.02	0.0938	20.03	0.0876
1min/10sec	21.17	0.0994	24.85	0.0991	34.95	0.0948
10sec/10sec	31.21	0.0993	43.26	0.0988	48.35	0.0986

Table 4.7. Baseline SVM experiments with constant test duration

	SVM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
4min/10sec	38.48	0.0977	36.65	0.0924	35.48	0.0925
1min/10sec	36.98	0.0969	31.09	0.0905	36.62	0.0874
10sec/10sec	36.92	0.0957	36.30	0.0930	37.60	0.0928

the GMM/UBM classifier has taken the lead from SVM in verification rates. The protocol that is most affected by test-duration change is the 4min case as expected, with a 7.5% accuracy loss in the worst case (GMM256). It should also be noted that GMM64 performs better than GMM256, contrary to GMM/UBM baseline. This implies that although more complex models might be better for modeling a given amount of training data, the verification performance may decrease if these models are used to test utterances of shorter durations.

SVM accuracy seems to be very much dependent on the test duration, too. For GMM16, EERs of three protocols are so close that it is both cost-effective and beneficial for one to use a training data of 10 seconds instead of 4 minutes for this particular situation. Figure 4.4 shows comparative DET curves of 4min/4min and 4min/10sec for both classifiers.

In Figure 4.4, as well as the other DET curves, an inconsistent structure is observed: The curves with 10sec test condition are almost linear, whereas the 4min test curves look piecewise and wavy. This phenomenon is believed to arise from test sample

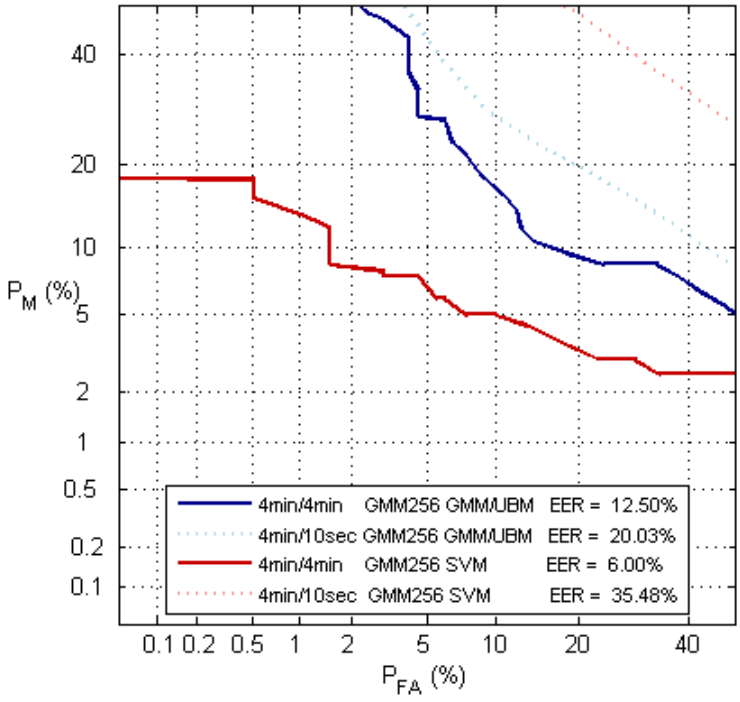

Figure 4.4. 4min/4min vs. 4min/10sec DET curves for GMM and SVM

imbalance. Since the total duration of test sessions are divided into smaller pieces of indicated durations, for the 4min case we have fewer test data and thus fewer samples to represent the miss and false alarms, which makes these curves seem unstable.

4.9. A Note on Session Variability

Having 12 sessions collected over a 2-year period, CSLU database is one of the most challenging ones in terms of session variability. We conducted a small experiment in order to find a measure on the degree of acoustic mismatch between sessions.

We use the GMM/UBM classifier setup, with 2 minutes of training data per speaker, taken from the first session. The "within-session" test group contains recordings of the other half (2 minutes) of the same session, while the "between-session" group has the same number of data from another session (session 7). Corresponding EER and minDCF values are as follows:

Table 4.8. Effect of Session Variability

	GMM/UBM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
within-session	12.33	0.0561	10.67	0.0547	12.80	0.0518
between-session	19.51	0.0832	22.76	0.0841	33.02	0.0875

The results in the table suggest that session variability affects both the EER and minDCF rates greatly, and its effects are more pronounced with higher mixture sizes. Section 6.4 specifically deals with the problem of reducing session variability effects for SVM classifiers.

5. EXPERIMENTS WITH LIMITED DATA

Verification results presented in Chapter 4 suggest that for moderately large amount of data (4min/4min), even the baseline performances of the classifiers are in somewhat acceptable error limits. However, for the 10sec/10sec protocol, the lowest EER we can achieve is 31.21%, which is far beyond the region of practical usage. In this chapter, we limit ourselves to the 10sec/10sec GMM16 case, to investigate deeply the methods that can be applied to increase the verification performance when working with limited data. The reader should be informed that most of these methods show a similar or better influence when applied to protocols other than this hypothetical worst case.

5.1. Changing the Adaptation Parameter and Type

Theoretical background on speaker adaptation methods were given in Section 3.3.3 and the following subsections. With these experiments, we aim to understand how changing the adaptation factor (τ) of MAP would affect the verification performance. We selected three adaptation factor values: $\tau = 14$ is the default value, which was also used in the baseline experiments. When $\tau = 0$, $\alpha = 1$ according to Equation 3.6 and so the speaker model means are not adapted, they are constructed independently using only that speaker's training data. To represent the other extreme case $\alpha \to 0$, we set $\tau = 10^5$ (Here, the mean values of each speaker were found out to differ after the third decimal digit). We also experimented MLLR adaptation instead of MAP, based on the fact that MLLR adaptation works better in adapting when the amount of training data is limited. Table 5.1 contains EER and minDCF values and Figure 5.1 demonstrates the DET curves of these experiments.

The first thing to be observed in Table 5.1 is that MLLR adaptation provides the lowest error rates, as expected. The drop in error rates is more distinct for the GMM/UBM classifier, by over 5%. For the SVM, using the default adaptation factor yields the best results in MAP, whereas for GMM/UBM, better values can be obtained

Table 5.1. Adaptation Changes for the 10sec/10sec protocol and GMM16 model

	MAP						MLLR	
	$\tau = 0$ ($\alpha = 1$)		$\tau = 14$ (Default)		$\tau = 10^5$ ($\alpha \to 0$)			
	EER	minDCF	EER	minDCF	EER	minDCF	EER	minDCF
GMM/UBM	30.22	0.0970	31.21	0.0993	28.03	0.0992	26.00	0.0991
SVM	37.53	0.0991	36.92	0.0957	40.80	0.0991	35.59	0.0987

by making the speaker models close to each other (i.e., $\alpha \to 0$). We had already observed in baseline experiments that GMM/UBM provides lower error rates than SVM for the 10sec/10sec protocol. This behavior does not change when we alter the adaptation parameters, although in some cases, lower minDCF may be encountered.

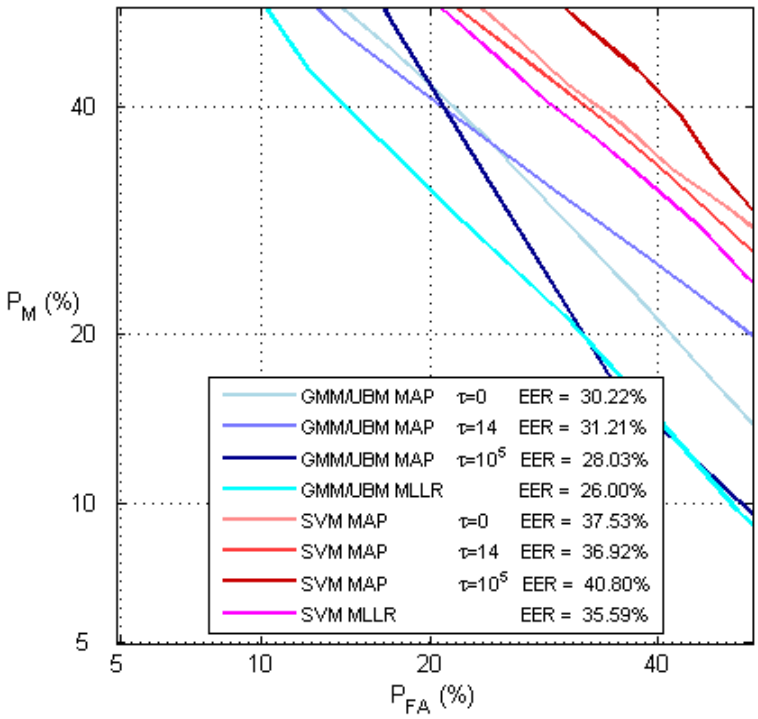

Figure 5.1. Effect of adaptation method for 10sec/10sec GMM16

5.2. Feature and Score Level Fusion

Considering the healing power of fusion strategies, we wonder if score-level fusion of MAP(default) and MLLR adapted systems would yield better results. Since SVM supervector classifier uses adaptation outputs as their input features, we also had the chance to combine MAP- and MLLR-adapted means in a single extended supervector,

Table 5.2. Score- and feature-level MAP(default)-MLLR fusion for limited data

	GMM/UBM-Score		SVM-Score		SVM-Feature	
	EER	minDCF	EER	minDCF	EER	minDCF
Linear	27.57	0.0992	34.28	0.0967	34.65	0.0970
LLR	25.09	0.0992	34.41	0.0965		

thus applying feature-level fusion. No scaling or normalization were applied in these experiments. We present the results of these three operations for linear and logistic linear regression fusion strategies in the following table.

It can be seen that only a little improvement can be achieved by combining MAP and MLLR features in the SVM setting, and the return is not higher than the one obtained by score fusion. In terms of combining different adaptation methods, the lowest EER we can get is a 3.5% relative change by GMM/UBM score fusion over the MLLR-only results.

5.3. Changing the Kernel

We now try to see the influence of kernel type on limited data SVM verification accuracy. All SVM experiments explained so far have used the linear kernel. For this section we apply the types presented in section 3.3.6.1, namely the RBF kernel, polynomial kernel, and superlinear kernel. For the RBF kernel, σ is chosen as $\sqrt{(KD/2)}$ where KD denotes the dimensionality of the supervector. For the polynomial kernel, we select the degree as $p = 3$. We note the results for both default MAP- and MLLR-adapted features for the 10sec/10sec protocol and GMM16.

Table 5.3 points out that the effect of kernel change applies differently for MAP and MLLR supervectors. Superlinear kernel, for instance, decreases the EER by 1% for the MLLR case, while increasing it slightly for MAP. Another observation is that applying a nonlinear function to map data to a higher dimensional space may not be an appropriate choice for this single-positive-example setup, as the results of RBF and polynomial kernels suggest. Figure 5.2 depicts the DET curves of these experiments.

Table 5.3. Changing the kernel type for 10sec/10sec GMM16

	Kernel Type							
	Linear		RBF		Polynomial		SuperLinear	
	EER	minDCF	EER	minDCF	EER	minDCF	EER	minDCF
MAP	36.92	0.0957	37.63	0.0960	39.71	0.0957	37.21	0.0962
MLLR	35.59	0.0987	35.47	0.0993	38.92	0.0984	34.57	0.0988

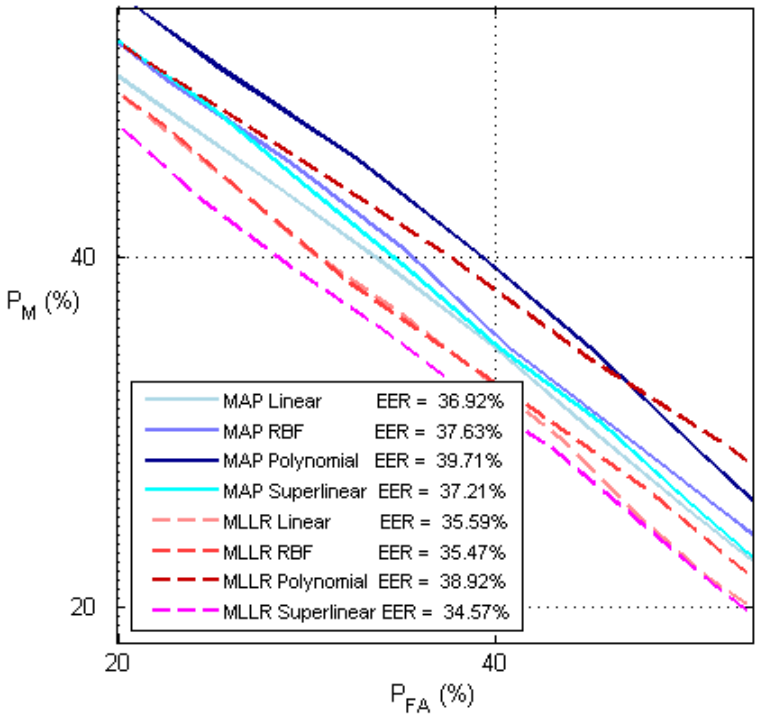

Figure 5.2. Effect of SVM kernel type on 10sec/10sec GMM16

5.4. Summary

This chapter deals with a challenging verification case, where training data is scarce (only one sample of 10 seconds) and test duration is short (10 seconds). The GMM/UBM classifier has a considerable superiority over the SVM. The gap between EERs is further widened if we use MLLR adaptation instead of MAP, a fact well known in theory. Combining MAP and MLLR adapted GMM/UBM scores under the LLR fusion leads to slightly increased verification accuracy. Other operations on the SVM setting, such as feature-level fusion and kernel function selection enhance results to some extent, but still does not reach the performance expressed by the GMM/UBM.

6. EXPERIMENTS WITH EXTENSIVE DATA

In contrast to the experiments in Chapter 5, we now investigate the practical lower error bounds of the speaker verification system, by using all six sessions for training, which corresponds to a duration of around 24 minutes per speaker.

6.1. GMM/UBM and SVM Behavior

GMM/UBM experiments were applied similarly to the baseline setup, which was presented in Section 4.5. For the SVM case, instead of using one single example for each speaker (positive class), we now use multiple supervectors, derived by dividing the training data into collections of 4 minutes, 1 minute and 10 seconds, respectively. Background supervectors are accordingly formed from the relevant data subset. For instance, for the 4min case we have 6 speaker supervectors against 120 background supervectors, whereas for 10sec, we have 144 positive samples against 2880 negatives. Testing is performed as usual, over our three main protocols. This time, probability output score is used for the SVM decision, instead of the customary $f(\mathbf{x})$ value. Table 6.1 and Table 6.2 show EER and minDCF values of these experiments.

Table 6.1. EER and minDCF values for extensive data GMM/UBM experiments

	GMM/UBM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(24min)/4min	11.59	0.0770	9.40	0.0610	6.71	0.0490
(24min)/1min	15.59	0.0914	11.73	0.0806	8.52	0.0600
(24min)/10sec	17.36	0.0984	14.27	0.0888	10.29	0.0758

Table 6.1 expresses the importance of testing duration over the verification performance. As the duration increases, the GMM/UBM classifier can make a better decision on whether the speech belongs to the ID it claims. We observe a relative EER decrease of up to 17.8% for a 6-times increase in duration (10sec $\rightarrow$ 1min), and about

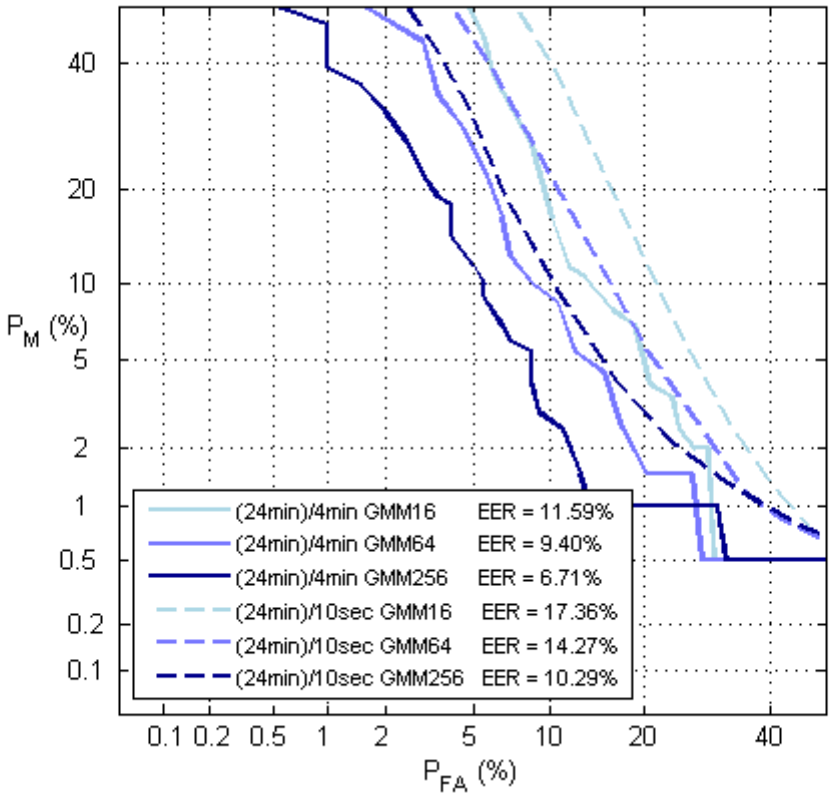

Figure 6.1. GMM/UBM extensive data experiments DET curve

35% decrease for a 24-times increase (10sec → 4min). The choice of whether requiring 4-times more data for an additional relative 50% decrease in error obviously depends on data availability, desired error tolerance and rapid decision expectancy for that specific application. We also note that these values are not the best naive verification rates to be achieved with 24 minutes of training data. It is very likely that higher order models (such as GMM512) would lead to more successful outcomes for each of these three testing durations.

Table 6.2. EER and minDCF values for extensive data SVM experiments

	SVM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(4minx6)/4min	7.10	0.0260	2.53	0.0090	2.50	0.0025
(1minx24)/1min	8.46	0.0362	4.80	0.0255	4.61	0.0249
(10secx144)/10sec	12.69	0.0592	10.82	0.0473	8.10	0.0394

Rows of Table 6.2 can be viewed as improved versions of the rows of Table 4.3, having training durations extended to 24 minutes. Comparing the results, we conclude that adding five more positive class samples for the 4 minute test case drops the EER from 6.00% to 2.50%. For the (10secx144)/10sec case, the downfall is more dramatic: from 37.60% to 8.10%. It can also be seen that increase in model size (and hence the SVM space dimensionality) increases performance for each of the three protocols, an occasion not observed for the 10sec/10sec case of Table 4.3.

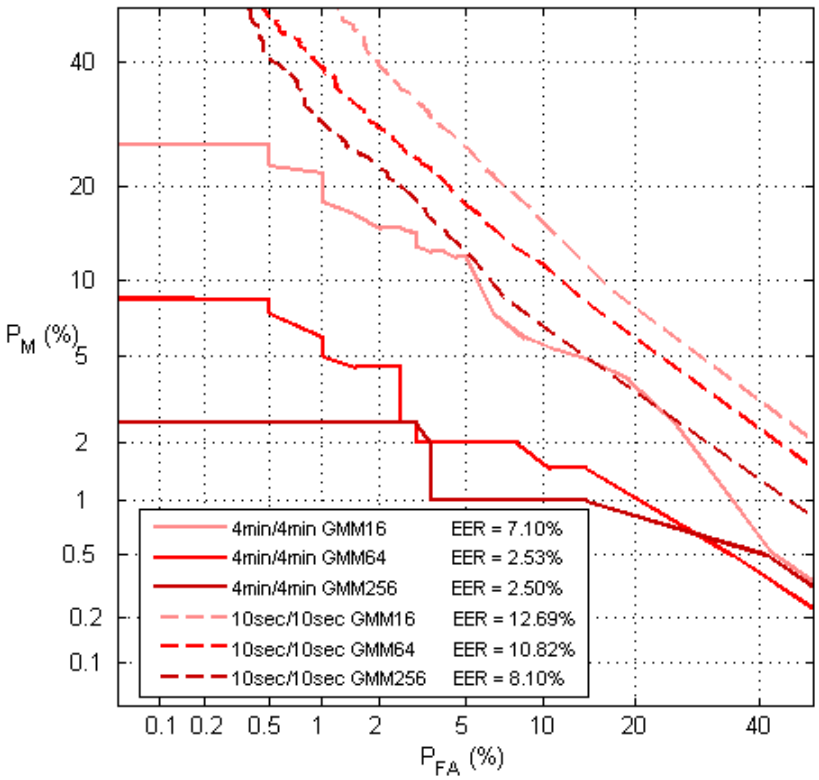

Figure 6.2. SVM extensive data experiments DET curve

A comparison of GMM/UBM and SVM accuracies for the best (24min/4min GMM256) and worst (24min/10sec GMM16) protocols is demonstrated in Figure 6.3. It is interesting to see that for 10sec case, although the EER of SVM is considerably lower than that of the GMM/UBM, one should select the GMM/UBM setup if it is desired to work in the low miss probability region (below 5%).

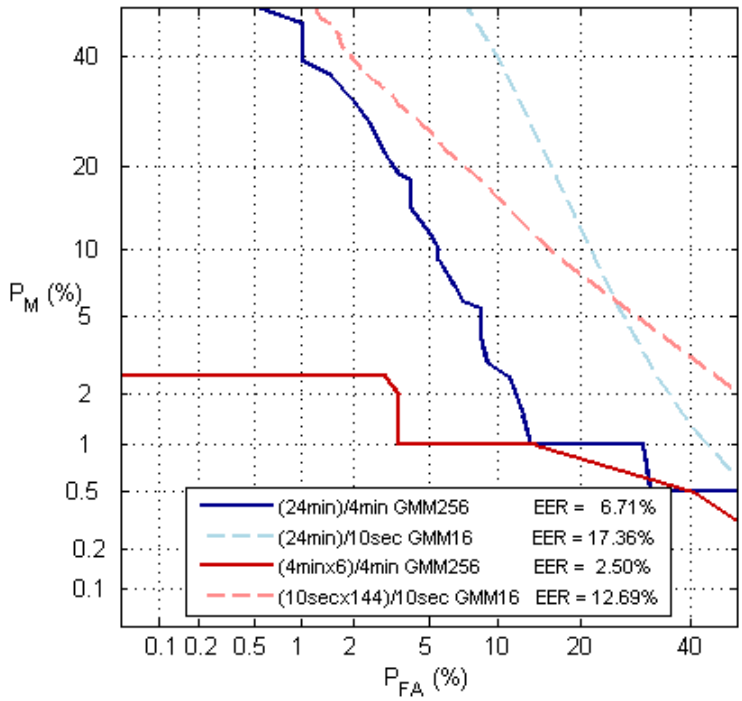

Figure 6.3. Comparison of GMM/UBM and SVM for extensive data

6.2. Probabilistic vs. Traditional Output Scores for SVM

We have stated that we chose the probabilistic output type for SVM experiments with extensive data, instead of the discriminating function value. Table 6.3 shows how the evaluation metrics would become if the same experiments in Table 6.2 were calculated using the $f(\mathbf{x})$ values.

As it can be seen, for the (1minx24)/1min and (10secx144)/10sec protocols, using the probability output generally yields lower error rates and minDCF values. The reason can be explained as follows: In these cases, the total duration is divided into smaller pieces, so we have more training samples for the same amount of 24 minutes. Since parameters of the function $p(y = 1|f^*(\mathbf{x}))$ in Equation 3.28 are determined by optimizing over the training samples, we can get more accurate estimates, which in turn decreases the error rates.

Table 6.3. Using $f(\mathbf{x})$ for extensive data SVM experiments

	SVM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(4minx6)/4min	7.04	0.0245	2.53	0.0110	1.67	0.0025
(1minx24)/1min	8.73	0.0378	5.03	0.0265	4.66	0.0270
(10secx144)/10sec	13.08	0.0588	10.70	0.0478	8.13	0.0403

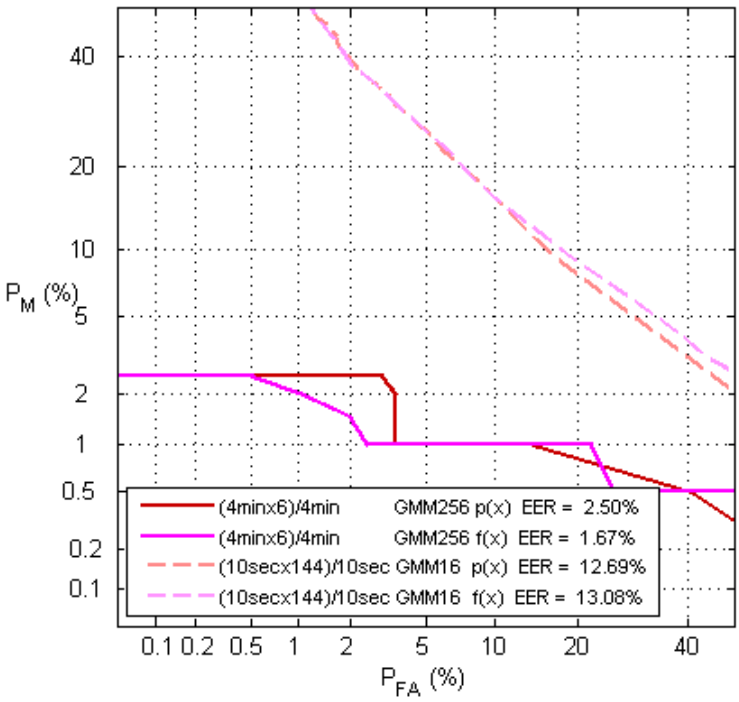

Figure 6.4. SVM output score comparison

6.3. Changing the Kernel

In Section 5.3 we commented on how applying different kernels would change the performance of SVMs. This time we repeat these experiments for the extensive data case, with three selected protocols and GMM sizes, shown in Table 6.4.

The results in the table seem mixed and rather confusing. Together they reveal the idea that changing kernel type acts on the verification accuracy unconformably for different protocols and adaptation types. For instance, the curative effect of superlinear kernel is visible in MAP-adapted (4minx6)/4min GMM16 and (10secx144)/10sec GMM16 cases, where an opposite behavior is observed for their MLLR-adapted ver-

Table 6.4. Changing the kernel type for extensive data

		Kernel Type							
		Linear		RBF		Polynomial		SuperLinear	
		EER	minDCF	EER	minDCF	EER	minDCF	EER	minDCF
MAP	(4minx6)/4min GMM16	7.10	0.0260	7.29	0.0285	9.30	0.0290	6.69	0.0240
	(4minx6)/4min GMM256	2.50	0.0025	2.50	0.0025	2.02	0.0040	2.53	0.0070
	(10secx144)/10sec GMM16	12.69	0.0592	12.38	0.0606	12.67	0.0631	12.03	0.0574
MLLR	(4minx6)/4min GMM16	4.18	0.0140	5.54	0.0190	7.64	0.0255	6.51	0.0205
	(4minx6)/4min GMM256	2.02	0.0090	3.00	0.0135	3.93	0.0130	2.50	0.0080
	(10secx144)/10sec GMM16	13.65	0.0628	12.91	0.0583	12.22	0.0607	11.84	0.0547

sions. RBF kernel shows its power in the (10secx144)/10sec GMM16 protocol for both adaptation types, whereas polynomial kernel exhibits the largest relative change by a 20% reduction in EER for the MAP-adapted (4minx6)/4min GMM256 setup.

6.4. Nuisance Attribute Projection

We have shown in Section 4.9 that session variability is one of the main reasons for low verification accuracy. To decrease channel variability effects, we apply the NAP method following the guidelines presented in Section 3.4.5.

The intersession variation matrix $\mathbf{M}$ of Equation 3.34 is computed using the training data. For each utterance, the corresponding low rank matrix $\mathbf{R}$ is used to project out the channel subspace. The dimensionality of the channel subspace is selected as $K = 40$. This value was seen to yield the lowest EER, after some preliminary tests. Table 6.5 shows NAP-applied SVM extensive results.

When compared with Table 6.2, we can conclude that NAP has a considerable effect on removing the channel/session variability phenomena. The improvement in accuracy is most pronounced in the (4minx6)/4min GMM256 protocol, with a 250% relative reduction in EER with only a slight increase in minDCF. However, it is observed that applying NAP does not heal the system for the (10secx144)/10sec case. This observation is consistent with [44], where NAP is applied on the 10s10s condition of NIST evaluations. Figure 6.5 depicts the DET curves of these experiments.

Table 6.5. EER and minDCF values after NAP on extensive data SVM experiments

	SVM NAP					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(4minx6)/4min	6.37	0.0160	2.26	0.0065	1.00	0.0030
(1minx24)/1min	7.76	0.0349	4.77	0.0235	4.04	0.0204
(10secx144)/10sec	14.98	0.0637	11.74	0.0510	8.94	0.0411

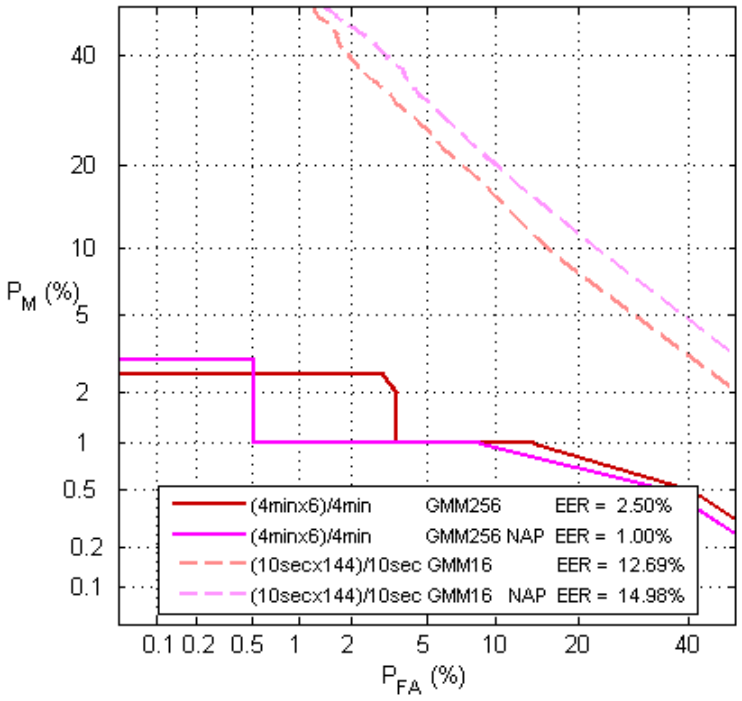

Figure 6.5. NAP performance ($K = 40$)

6.5. Minority Oversampling

As explained in Section 3.3.6.3, the SVM setups we use suffer from data imbalance. We utilize the SMOTE algorithm and generate samples from the minority class, to hopefully enlarge the minority class region by creating more positive support vectors. The number of neighbors in this algorithm is selected as $n = 3$ and the oversampling rate is chosen as 20, since for each one-against-background SVM, we train samples from one speaker against a collection of 20 background speakers. Table 6.6 displays outcomes of the SMOTE implementation.

Table 6.6. EER and minDCF values after SMOTE on extensive data SVM experiments

	SVM SMOTE					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(4minx6)/4min	8.07	0.0240	2.68	0.0110	1.25	0.0035
(1minx24)/1min	10.83	0.0383	5.34	0.0282	4.61	0.0249
(10secx144)/10sec	12.69	0.0592	10.75	0.0477	8.10	0.0394

Comparing Table 6.6 with Table 6.2, we see that SMOTE can improve error rates only when the dimensionality of space (GMM256), or the number of minority samples (10sec) is high. For the opposite case, the artificial support vectors introduced by applying the algorithm seem to degrade the naturalness of the decision boundary, which results in high EER. There is also a probability that the newly introduced samples may not contribute to shaping the boundary, as in the case of (1minx24)/1min GMM256 and (10secx144)/10sec GMM16. Two SMOTE trials can be viewed in Figure 6.6.

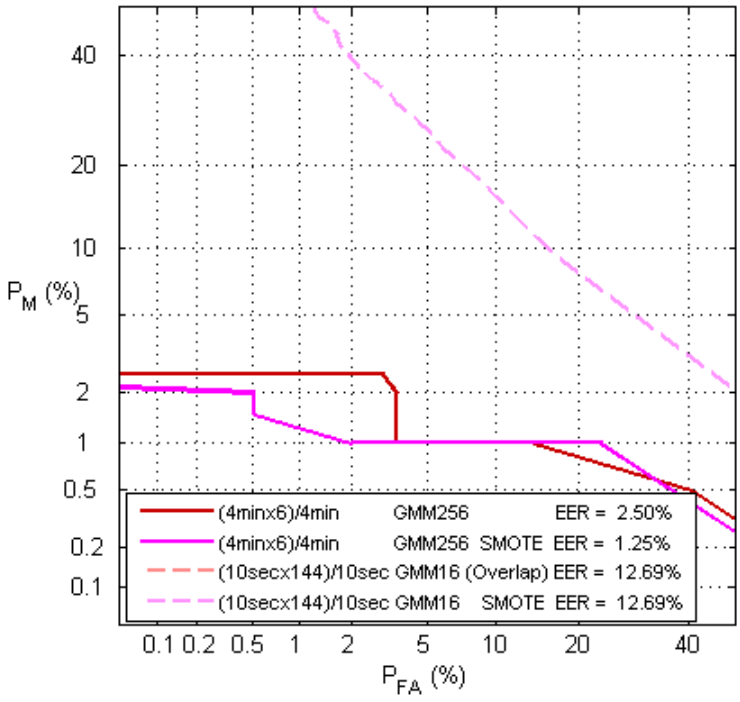

Figure 6.6. SMOTE performance

6.6. Extensive Data with Constant Training Partitioning

Results placed in the rows of Table 6.2 provide a consistent evaluation of our three baseline protocols being adapted to 24min case. However, these numbers do not imply how verification performance changes when test duration is altered, because the way training duration (24 minutes) is distributed among the supervectors is not the same. To better comment on how test durations affect SVM decisions, we fix the training data partitioning to (4minx6) case and repeat the experiments with changing test conditions. We present the results in Table 6.7.

Table 6.7. EER and minDCF values for extensive data SVM experiments with constant training partitioning

	SVM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(4minx6)/4min	7.10	0.0260	2.53	0.0090	2.50	0.0025
(4minx6)/1min	17.44	0.0711	12.15	0.0527	11.61	0.0546
(4minx6)/10sec	31.30	0.0985	30.34	0.0965	32.59	0.0988

A fair interpretation of Table 6.7 can be made by contrasting it to Table 6.1. An interesting relation is revealed: Although SVM creates the best results for 4min test case, GMM/UBM classifier starts to outperform SVM as the test duration is decreased. This dominance is more evident with increasing model sizes. For instance, in 10sec condition with GMM256, there is over 3-times improvement for the EER, and a relative decrease of 25% for the minDCF. In Figure 6.7, GMM/UBM and SVM DET curves are shown for 4min GMM256 and 10sec GMM16 cases.

6.7. Extensive Data with Constant Test Duration

Contrary to Section 6.6, we now try to find out how this flexible nature of SVM training would be compared under a fixed constant test duration of 10 seconds. EER and minDCF values of these trials are presented in Table 6.8.

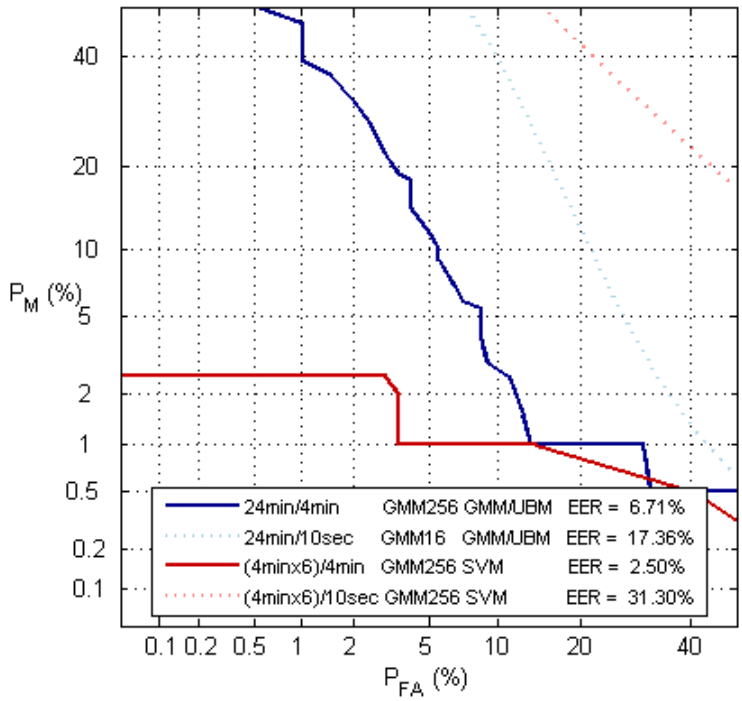

Figure 6.7. Comparison of GMM/UBM and SVM over constant training data partitioning

Values in Table 6.8 suggest another intriguing idea: If the testing duration is limited and fixed, it is better to train the SVM by dividing the training data into smaller segments which has an equivalent duration to that of the test recordings, than to use longer segments. The reason might be that supervectors composed of equal durations (10sec, in our case), hold an equivalent range of information, although it might not be the best way to represent the speaker's vocal characteristics. Dividing into smaller chunks also makes the supervectors large in quantity, so that the minority

Table 6.8. EER and minDCF values for extensive data SVM experiments with constant test duration of 10 seconds

	SVM					
	GMM16		GMM64		GMM256	
	EER	minDCF	EER	minDCF	EER	minDCF
(4minx6)/10sec	31.30	0.0985	30.34	0.0965	32.59	0.0988
(1minx24)/10sec	18.11	0.0766	16.20	0.0723	21.55	0.0900
(10secx144)/10sec	12.69	0.0592	10.82	0.0473	8.10	0.0394

class could be better represented. Figure 6.8 compares GMM256 models for changing training partitions.

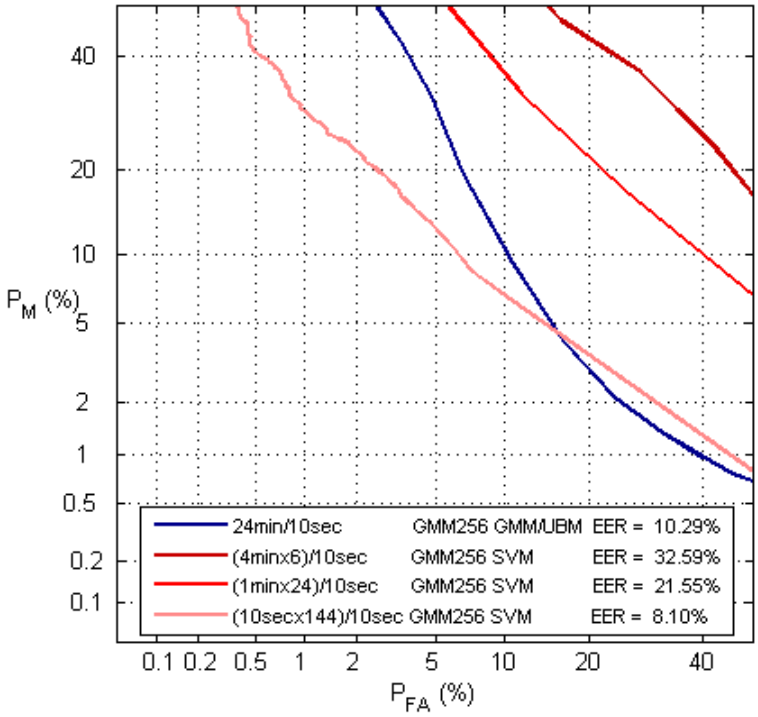

Figure 6.8. Comparison of training data partitions under 10sec constant test duration

6.8. Summary

This chapter investigates the case where large amount of data is available for training. SVM provides results superior to GMM/UBM for each of the three main protocols. Lower EERs can be obtained if probability scores are used for decision, instead of the discriminative function values. Experiments reveal that the kernel which yields the lowest error rates depends on the setup and methods used. Applying NAP has the greatest improvement on verification performance, except when working with the 10sec case. Oversampling with SMOTE may lead to higher accuracies, depending on the distribution of new samples in the feature space.

Regardless of the amount of training data, verifying an identity based on as short an utterance as possible is an important task in speaker verification. Comparing all tests conducted so far with 10 seconds of test duration would help us gain more insight

on the short duration testing problem. Table 6.9 combines the results of relevant tests from Tables 4.6, 4.7, 6.1 and 6.8.

Table 6.9. Verification performance comparison for all 10sec tests

		GMM16		GMM64		GMM256	
		EER	minDCF	EER	minDCF	EER	minDCF
GMM/UBM	24min/10sec	17.36	0.0984	14.27	0.0888	10.29	0.0758
	4min/10sec	19.49	0.0995	18.02	0.0938	20.03	0.0876
	1min/10sec	21.17	0.0994	24.85	0.0991	34.95	0.0948
	10sec/10sec	31.21	0.0993	43.26	0.0988	48.35	0.0986
SVM	(10secx144)/10sec	12.69	0.0592	10.82	0.0473	8.10	0.0394
	(1minx24)/10sec	18.11	0.0766	16.20	0.0723	21.55	0.0900
	(4minx6)/10sec	31.30	0.0985	30.34	0.0965	32.59	0.0988
	4min/10sec	38.48	0.0977	36.65	0.0924	35.48	0.0925
	1min/10sec	36.98	0.0969	31.09	0.0905	36.62	0.0874
	10sec/10sec	36.92	0.0957	36.30	0.0930	37.60	0.0928

Comparing the extended data GMM/UBM verification performance with those of the baseline experiments, we conclude that when the training data is increased for a factor of 6 (4 minutes to 24 minutes), we obtain a relative decrease of up to 50% in EER. Similarly for the SVM setup, adding new samples to the supervector space helps: The decrease in the error rate is about 5% by adding 5 more samples for the 4min training, whereas it reaches 28% when we add 143 more samples in the 10sec case.

7. CONCLUSION

In this study, we aimed at text-independent speaker verification problem and investigated the performance of GMM/UBM and SVM supervector classifiers under changing data amounts and model complexities. The experiments are repeated for three training/testing data durations (4min, 1min, 10sec), and three GMM mixture component sizes (16, 64, 256). We tried to understand the practicality of the verification system by observing both the theoretical limits of accuracy in an extended data application, and the worst case scenario in a limited data case. Besides altering data quantity, we also performed tests to see how the system would react to changes in model setup parameters by modifying the adaptation method and kernel type. Finally, we checked if session variability compensation and artificial data generation techniques help decrease the error rates.

Based on the obtained results, the following conclusions can be drawn:

- Support vector machine is a successful classifier that outperforms GMM/UBM in both limited and extensive data verification cases where the training and test samples are generated using the same recording durations. If testing durations are shorter than training, however, GMM/UBM may yield lower error rates.
- GMM supervector is an appropriate representation of speaker characteristics in SVM work space. It combines the generalization abilities of GMMs with the discriminative representation capabilities of SVMs.
- The type of adaptation affects verification performance of both GMM/UBM and SVM classifiers. MLLR adaptation works better if training data is limited, as in our 10sec case. Relevance factor of MAP adaptation also influences error rates, but their influence is not similar for the two classifiers.
- The choice of an appropriate kernel function is another important decision in constructing the SVMs. The effect of kernel on the verification accuracy varies according to sample size, model complexity, and adaptation type. Inherent parameters of the kernel should be optimized to obtain the best results.

- Session variability is one of the most challenging issues against obtaining robust verification performance. Nuisance attribute projection (NAP) is an effective method on mitigating influence of variability. However, it needs multiple training sessions to estimate potential channel changes, and is not successfully applied for short duration data conditions.
- Although oversampling the minority class with SMOTE is a reasonable method to prevent the class imbalance problem, it may act positively or negatively on the results, as it is hard to predict whether the artificially generated samples would contribute to create a better decision boundary in the high dimensional space.

The number of user and impostor test attempts has been found to be a crucial factor in computing the evaluation metrics. For instance, since SVM baseline experiments use only a single training example for the speaker class, the resulting decision boundary eventually favors the majority class; in other words, it tends to classify all test samples as impostor attempts. Therefore, it can be deducted that an increase in the number of impostor tests would mean lower error rates for this type of a classifier. In our experiments, we tried to equalize the number of legal and illegal attempts to the system.

This issue also presents the importance of setting a common usage scenario, when comparing two speaker verification systems. The Speaker Recognition Evaluations by NIST thus constitutes a valuable framework for institutions to compare their algorithms and collaborate on the subject.

The accuracies for cases where lots of training data are involved allow verification systems to be used in practical applications such as broadcast news and meeting annotations, and forensic decision making. However, the results for 10sec case reveal that there is still much to do to enhance speaker verification performance of limited data systems.

In light of these findings, this study can be extended towards obtaining a speaker verification system for limited data and short duration testing applications. Meth-

ods that increase robustness of the system against session variations should be explored. Further channel variability reduction, automatized parameter selection and fusion methods could be investigated for this purpose. Measuring the sensitivity of the system to voice transformation attacks is also an issue of interest. An ultimate objective would be integrate this setup into a multimodal authentication system, which not only combines the outputs of different modalities, but also uses the relationships among these to finally arrive at a more accurate decision.

REFERENCES

1. Kung, S., M. Mak, and S. Lin, *Biometric authentication: a machine learning approach*, Prentice Hall Press, Upper Saddle River, NJ, USA, 2004.

2. Reynolds, D., "An overview of automatic speaker recognition technology", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '02)*, Vol. 4, pp. IV–4072–IV–4075, 2002.

3. Selvi, M., *Effects of root cepstral coefficients on speaker recognition performance over telephone channels*, Master's thesis, Boğaziçi University, 2002.

4. Jin, Q., A. Toth, A. Black, and T. Schultz, "Is voice transformation a threat to speaker identification?", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '08)*, pp. 4845–4848, April 2008.

5. "Applications for AMI technologies", Technical report, AMI Consortium, 2006.

6. Bimbot, F., J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-Garcia, D. Petrovska-Delacretaz, and D. A. Reynolds, "A tutorial on text-independent speaker verification", *EURASIP Journal on Applied Signal Processing*, Vol. 4, pp. 430–451, 2004.

7. Campbell, J. P., "Speaker recognition: a tutorial", *Proceedings of the IEEE*, Vol. 85, No. 9, pp. 1437–1462, 1997.

8. Reynolds, D. A. and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models", *IEEE Transactions on Speech and Audio Processing*, Vol. 3, No. 1, pp. 72–83, 1995.

9. Bonastre, J.-F., N. Scheffer, C. Fredouille, and D. Matrouf, "NIST'04 speaker recognition evaluation campaign: new LIA speaker detection plateform based on

ALIZE toolkit", *Proceedings of NIST Speaker Recognition Evaluation (SRE '04)*, June 2004.

10. Liu, M., B. Dai, Y. Xie, and Z. Yao, "Improved GMM-UBM/SVM for speaker verification", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '06)*, Vol. 1, pp. I–925 – I–928, May 2006.

11. Dehak, N. and G. Chollet, "Support vector GMMs for speaker verification", *Proc. IEEE Odyssey 2006: The Speaker and Language Recognition Workshop*, pp. 1–4, June 2006.

12. Mayoue, A., "Reference system based on speech modality ALIZE/LIA-RAL", Technical report, GET-INT, 2008.

13. Schmidt, M. and H. Gish, "Speaker identification via support vector classifiers", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '96)*, pp. 105–108, IEEE Computer Society, Washington, DC, USA, 1996.

14. Bengio, S. and J. Mariethoz, "Learning the decision function for speaker verification", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '01)*, Vol. 1, pp. 425–428, 2001.

15. Zhao, X., Y. Dong, H. Yang, J. Zhao, and H. Wang, "SVM-based speaker verification by location in the space of reference speakers", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '07)*, Vol. 4, pp. IV–281–IV–284, April 2007.

16. Campbell, W., D. Sturim, D. Reynolds, and A. Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '06)*, Vol. 1, pp. I–97–I–100, May 2006.

17. Wan, V. and S. Renals, "SVMSVM: Support vector machine speaker verification

methodology", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '03)*, Vol. 2, pp. II–221–4, April 2003.

18. Moreno, P. J. and P. P. Ho, "A new SVM approach to speaker identification and verification using probabilistic distance kernels", *Proc. EUROSPEECH'03*, pp. 2965–2968, 2003.

19. Louradour, J. and K. Daoudi, "SVM speaker verification using a new sequence kernel", *Proc. European Signal Processing Conference (EUSIPCO '05)*, 2005.

20. Campbell, W., J. Campbell, D. Reynolds, E. Singer, and P. Torres-Carrasquillo, "Support vector machines for speaker and language recognition", *Computer Speech & Language*, Vol. 20, No. 2-3, pp. 210–229, 2006.

21. Dehak, R., N. Dehak, P. Kenny, and P. Dumouchel, "Comparison between factor analysis and GMM support vector machines for speaker verification", *Proc. IEEE Odyssey 2008: The Speaker and Language Recognition Workshop*, January 2008.

22. Campbell, W., D. Sturim, and D. Reynolds, "Support vector machines using GMM supervectors for speaker verification", *IEEE Signal Processing Letters*, Vol. 13, No. 5, pp. 308–311, May 2006.

23. Dehak, R., N. Dehak, P. Kenny, and P. Dumouchel, "Linear and non linear kernel GMM supervector machines for speaker verification", *Proc. 8th Annual Conference of the International Speech Communication Association (INTERSPEECH '07)*, August 2007.

24. Fauve, B., D. Matrouf, N. Scheffer, J.-F. Bonastre, and J. Mason, "State-of-the-art performance in text-independent speaker verification through open-source software", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15, No. 7, pp. 1960–1968, Sept. 2007.

25. Dehak, R., N. Dehak, P. Kenny, and P. Dumouchel, "Kernel combination for SVM

speaker verification", *Proc. IEEE Odyssey 2008: The Speaker and Language Recognition Workshop*, January 2008.

26. Solomonoff, A., W. Campbell, and I. Boardman, "Advances in channel compensation for SVM speaker recognition", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '05)*, Vol. 1, pp. 629–632, 2005.

27. Auckenthaler, R., M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems", *Digital Signal Processing*, Vol. 10, No. 1-3, pp. 42–54, January 2000.

28. Campbell, W. M., D. A. Reynolds, and J. P. Campbell, "Fusing discriminative and generative methods for speaker recognition: Experiments on switchboard and NFI/TNO field data", *Proc. IEEE Odyssey 2004: The Speaker and Language Recognition Workshop*, pp. 41–44, May 2004.

29. Campbell, W., D. Sturim, W. Shen, D. Reynolds, and J. Navratil, "The MIT-LL/IBM 2006 speaker recognition system: High-performance reduced-complexity recognition", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '07)*, Vol. 4, pp. IV–217–IV–220, April 2007.

30. Solomonoff, A., C. Quillen, and W. Campbell, "Channel compensation for SVM speaker recognition", *Proc. IEEE Odyssey 2004: The Speaker and Language Recognition Workshop*, pp. 57–62, May 2004.

31. P.Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Factor analysis simplified", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '05)*, pp. 637–640, 2005.

32. Vogt, R. and S. Sridharan, "Experiments in session variability modelling for speaker verification", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '06)*, Vol. 1, pp. I–897–I–900, May 2006.

33. McLaren, M., R. Vogt, and S. Sridharan, *SVM speaker verification using session variability modelling and GMM supervectors*, pp. 1077–1084, LNCS 4642, Springer-Verlag Berlin Heidelberg, 2007.

34. Zhao, X., Y. Dong, H. Yang, J. Zhao, L. Lu, and H. Wang, "Nonlinear kernel nuisance attribute projection for speaker verification", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '08*, pp. 4125–4128, April 2008.

35. Burget, L., P. Matejka, P. Schwarz, O. Glembek, and J. Cernocky, "Analysis of feature extraction and channel compensation in a GMM speaker recognition system", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15, No. 7, pp. 1979–1986, Sept. 2007.

36. Vogt, R. and S. Sridharan, "Explicit modelling of session variability for speaker verification", *Computer Speech & Language*, Vol. 22, No. 1, pp. 17–38, 2008.

37. Ferrer, L., K. Sonmez, and S. Kajarekar, "Class-dependent score combination for speaker recognition", *Proc. EUROSPEECH'05*, pp. 2173–2176, 2005.

38. Stolcke, A., L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, "MLLR transforms as features in speaker recognition", *Proc. EUROSPEECH'05*, pp. 2425–2428, 2005.

39. Stolcke, A., E. Shriberg, L. Ferrer, S. Kajarekar, K. Sonmez, and G. Tur, "Speech recognition as feature extraction for speaker recognition", *Proc. SAFE 2007: Workshop on Signal Processing Applications for Public Security and Forensics*, pp. 39–43, 2007.

40. Mak, M.-W., R. Hsiao, and B. Mak, "A comparison of various adaptation methods for speaker verification with limited enrollment data", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '06)*, Vol. 1, pp. I–929–I–932, May 2006.

41. Xie, Y., B. Dai, Z. Yao, and M. Liu, "Kurtosis normalization in feature space for robust speaker verification", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '06)*, Vol. 1, pp. I–117–I–120, May 2006.

42. Vogt, R. J., S. Sridharan, and M. W. Mason, "Making confident speaker verification decisions with minimal speech", *Proc. 9th Annual Conference of the International Speech Communication Association (Interspeech '08)*, September 2008.

43. Vogt, R. J., C. J. Lustri, and S. Sridharan, "Factor analysis modelling for speaker verification with short utterances", *Proc. IEEE Odyssey 2008: The Speaker and Language Recognition Workshop*, 2008.

44. Fauve, B., N. W. D. Evans, N. R. Pearson, J.-F. Bonastre, and J. S. D. Mason, "Influence of task duration in text-independent speaker verification", *Proc. Interspeech '07*, pp. 794–797, 2007.

45. *The NIST year 2008 speaker recognition evaluation plan*, 2008.

46. Kim, H.-G., E. Berdahl, N. Moreau, and T. Sikora, "Speaker recognition using MPEG-7 descriptors", *Proc. Eurospeech '03*, 2003.

47. Özgür Devrim Orman, *Frequency analysis of speaker identification performance*, Master's thesis, Boğaziçi University, 2000.

48. Ganchev, T., N. Fakotakis, and G. Kokkinakis, "Comparative evaluation of various MFCC implementations on the speaker verification task", *Proc. 10th International Conference on Speech and Computer (SPECOM '05)*, 2005.

49. Reynolds, D. A., T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models", *Digital Signal Processing*, Vol. 10, No. 1-3, pp. 19–41, 2000.

50. Lee, C.-H., C.-H. Lin, and B.-H. Juang, "A study on speaker adaptation of the parameters of continuous density hidden Markov models", *IEEE Transactions on*

Signal Processing, Vol. 39, No. 4, pp. 806–814, April 1991.

51. Mokbel, C., H. Mokbel, R. Blouet, and G. Aversano, *BECARS library and tools for speaker verification*, 2008, `http://www.tsi.enst.fr/becars/index.php`.

52. Huang, X., A. Acero, and H.-W. Hon.

53. Leggetter, C. J. and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models", *Computer Speech & Language*, Vol. 9, No. 2, pp. 171 – 185, 1995.

54. Jurafsky, D. and J. H. Martin, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.

55. Huggins-Daines, D., "Speaker adaptation in Sphinx 3.x and CALO", PowerPoint presentation, Carnegie Mellon University.

56. Scholkopf, B. and A. J. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond*, MIT Press, Cambridge, MA, USA, 2001.

57. Milgram, J., M. Cherier, and R. Sabourin, "One-against-one or one-against-all: Which one is better for handwriting recognition with SVMs?", *Proc. 10th International Workshop on Frontiers in Handwriting Recognition*, October 2006.

58. Fine, S., J. Navratil, and R. Gopinath, "A hybrid GMM/SVM approach to speaker identification", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '01)*, Vol. 1, pp. 417–420, 2001.

59. Akbani, R., S. Kwek, and N. Japkowicz, *Applying support vector machines to imbalanced datasets*, Vol. 3201 of *Lecture Notes in Computer Science*, pp. 39–50, Springer, 2004.

60. Wang, B. X. and N. Japkowicz, *Boosting support vector machines for imbalanced*

data sets, Vol. 4994 of *Lecture Notes in Computer Science*, pp. 38–47, Springer, 2008.

61. Yuan, J., J. Li, and B. Zhang, "Learning concepts from large scale imbalanced data sets using support cluster machines", Nahrstedt, K., M. Turk, Y. Rui, W. Klas, and K. Mayer-Patel (editors), *ACM Multimedia*, pp. 441–450, ACM, 2006.

62. Veropoulos, K., C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines", *Proc. International Joint Conference on AI*, pp. 55–60, 1999.

63. Chawla, N., K. Bowyer, L. Hall, and W. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique", *Journal of Artificial Intelligence Research*, Vol. 16, pp. 321–357, 2002.

64. Platt, J. C., "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods", *Advances in Large Margin Classifiers*, pp. 61–74, MIT Press, 1999.

65. Lin, H.-T. and R. C. Weng, "A note on Platt's probabilistic outputs for support vector machines", 2003.

66. Martin, A., G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET curve in assessment of detection task performance", *Proc. Eurospeech '97*, pp. 1895–1898, 1997.

67. Fauve, B., "An introduction to the speaker verification task", PowerPoint presentation, University of Wales Swansea.

68. Li, K.-P. and J. Porter, "Normalizations and selection of speech segments for speaker recognition scoring", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '88*, Vol. 1, pp. 595–598, Apr 1988.

69. Matejka, R., L. Burget, P. Schwarz, O. Glembek, M. Karafiat, F. Grezl, J. Cer-

nocky, D. van Leeuwen, N. Brummer, and A. Strasheim, "STBU system for the NIST 2006 speaker recognition evaluation", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '07)*, Vol. 4, pp. IV–221–IV–224, April 2007.

70. Brummer, N., L. Burget, J. Cernocky, O. Glembek, F. Grezl, M. Karafiat, D. van Leeuwen, P. Matejka, P. Schwarz, and A. Strasheim, "Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15, No. 7, pp. 2072–2084, Sept. 2007.

71. CSLU, "CSLU: Speaker recognition version 1.1", 2006, Linguistic Data Consortium, Philadelphia.

72. "HTK Speech Recognition Toolkit", `http://htk.eng.cam.ac.uk/`.

73. "UNIANAL Universal Speech Analysis and Synthesis", `http://speech.fit.vutbr.cz/files/software/unianal/unianal.tar.gz`.

74. Mayoue, A., "Reference system based on speech modality BECARS/HTK", Technical report, GET-INT, 2008.

75. Bailly-Bailliére, E., S. Bengio, F. Bimbot, M. Hamouz, J. Kittler, J. Mariéthoz, J. Matas, K. Messer, V. Popovici, F. Porée, B. Ruiz, and J.-P. Thiran, "The BANCA database and evaluation protocol", *4th Int. Conf. on Audio- and Video-Based Biometric Person Authentication (AVBPA '03)*, Vol. 2688 of *Lecture Notes in Computer Science*, pp. 625–638, Springer, January 2003.

76. Chang, C.-C. and C.-J. Lin, *LIBSVM: A library for support vector machines*, 2001, `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.