VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

Extensions to Probabilistic Linear Discriminant Analysis for Speaker Recognition

Rozšíření pro pravděpodobnostní lineární diskriminační analýzu
v rozpoznávání mluvčího

DISERTAČNÍ PRÁCE
PHD THESIS

AUTOR PRÁCE Ing. OLDŘICH PLCHOT
AUTHOR

VEDOUCÍ PRÁCE Ing. LUKÁŠ BURGET, Ph.D.
SUPERVISOR

BRNO 2014

Abstract

This thesis deals with probabilistic models for automatic speaker verification. In particular, the Probabilistic Linear Discriminant Analysis (PLDA) model, which models i–vector representation of speech utterances, is analyzed in detail. The thesis proposes extensions to the standard state-of-the-art PLDA model. The newly proposed Full Posterior Distribution PLDA models the uncertainty associated with the i–vector generation process. A new discriminative approach to training the speaker verification system based on the PLDA model is also proposed.

When comparing the original PLDA with the model extended by considering the i–vector uncertainty, results obtained with the extended model show up to 20% relative improvement on tests with short segments of speech. As the test segments get longer (more than one minute), the performance gain of the extended model is lower, but it is never worse than the baseline. Training data are, however, usually available in the form of segments which are sufficiently long and therefore, in such cases, there is no gain from using the extended model for training. Instead, the training can be performed with the original PLDA model and the extended model can be used if the task is to test on the short segments.

The discriminative classifier is based on classifying pairs of i–vectors into two classes representing target and non-target trials. The functional form for obtaining the score for every i–vector pair is derived from the PLDA model and training is based on the logistic regression minimizing the cross-entropy error function between the correct labeling of all trials and the probabilistic labeling proposed by the system. The results obtained with discriminatively trained system are similar to those obtained with generative baseline, but the discriminative approach shows the ability to output better calibrated scores. This property leads to a better actual verification performance on an unseen evaluation set, which is an important feature for real use scenarios.

Keywords

Speaker Recognition, Gaussian Mixture Model, Subspace Modeling, i–vector, Probabilistic Linear Discriminant Analysis, Discriminative Training

Bibliographic citation

Oldřich Plchot: Extensions to Probabilistic Linear Discriminant Analysis for Speaker Recognition, **Doctoral thesis**, **Brno**, **Brno University of Technology, Faculty of Information Technology**, 2014

Abstrakt

Tato práce se zabývá pravděpodobnostními modely pro automatické rozpoznávání řečníka. Podrobně analyzuje zejména pravděpodobnostní lineární diskriminační analýzu (PLDA), která modeluje nízkodimenzionální reprezentace promluv ve formě i–vektorů. Práce navrhuje dvě rozšíření v současnosti požívaného PLDA modelu. Nově navržený PLDA model s plným posteriorním rozložením modeluje neurčitost při generování i–vektorů. Práce také navrhuje nový diskriminativní přístup k trénování systému pro verifikaci řečníka, který je založený na PLDA.

Pokud srovnáváme původní PLDA s modelem rozšířeným o modelování neurčitosti i–vektorů, výsledky dosažené s rozšířeným modelem dosahují až 20% relativního zlepšení při testech s krátkými nahrávkami. Pro delší testovací segmenty (více než jedna minuta) je zisk v přesnosti menší, nicméně přesnost nového modelu není nikdy menší než přesnost výchozího systému. Trénovací data jsou ale obvykle dostupná ve formě dostatečně dlouhých segmentů, proto v těchto případech použití nového modelu neposkytuje žádné výhody při trénování. Při trénování může být použit původní PLDA model a jeho rozšířená verze může být využita pro získání skóre v případě, kdy se bude provádět testování na krátkých segmentech řeči.

Diskriminativní model je založen na klasifikaci dvojic i–vektorů do dvou tříd představujících oprávněný a neoprávněný soud (target a non-target trial). Funkcionální forma pro získání skóre pro každý pár je odvozena z PLDA a trénování je založeno na logistické regresi, která minimalizuje vzájemnou entropii mezi správným označením všech soudů a pravděpodobnostním označením soudů, které navrhuje systém. Výsledky dosažené s diskriminativně trénovaným klasifikátorem jsou podobné výsledkům generativního PLDA, ale diskriminativní systém prokazuje schopnost produkovat lépe kalibrované skóre. Tato schopnost vede k lepší skutečné přesnosti na neviděné evaluační sadě, což je důležitá vlastnost pro reálné použití.

Klíčová slova

rozpoznávání mluvčího, směs gaussovských rozložení, modelování v podprostoru parametrů, i–vektor, pravděpodobnostní lineární diskriminační analýza, diskriminativní trénování

Bibliografická citace

Oldřich Plchot: Rozšíření pro pravděpodobnostní lineární diskriminační analýzu v rozpoznávání mluvčího, **Disertační práce**, **Brno**, **Vysoké učení technické v Brně, Fakulta informačních technologií**, 2014

Declaration of Originality

I hereby declare that this thesis and the work reported herein was composed by and originated entirely from me. The work has been supervised by Doc. Dr. Ing. Jan Černocký and Ing. Lukáš Burget, Ph.D. Information derived from the published and unpublished work of others has been acknowledged in the text and references are given in the list of sources. Some of the reported systems were created by the members of the BUT Speech@FIT research group or in cooperation with third parties (Agnitio, BBN Technologies, CRIM, SRI International).

Prohlášení

Prohlašuji, že jsem tuto disertační práci vypracoval samostatně pod vedením Doc. Dr. Ing. Jana Černockého a Ing. Lukáše Burgeta, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal. Některé systémy použité v práci byly vytvořeny členy výzkumné skupiny BUT Speech@FIT samostatně nebo ve spolupráci se třetími stranami (Agnitio, BBN Technologies, CRIM, SRI International).

Acknowledgments

I would like to thank Lukáš Burget and Honza Černocký for being excellent supervisors and friends, who always passionately helped me with any kind of problem. Endless discussions with Lukáš on various topics were always an inspiration to other work as well as a welcomed distraction. It was an honor and an excellent experience to work under the supervision of such great people.

I have to thank to all present and past members of the BUT Speech@FIT research group who always form a great team and achieve great things and who are always there to help. I especially want to thank to people with whom I had the honor to closely cooperate: Sandro Cumani, Pavel Matějka, Ondra Glembek, Mehdi Soufifar, Martin Karafiát, David Martínez González, Mireia Diez, Karel Veselý, Petr Schwarz and Tomáš Mikolov.

I would like to thank to Niko Brümmer who was my inspiration during our work on ABC submissions and from whom I learned how to build and manage the huge NIST-like (and bigger) systems. Big thanks to Niko also for sharing his scripts and other tools, which were used in this work and which are a great contribution to the whole community.

Thanks to all colleagues and friends with whom I was working on various projects and who helped me to gain so much experience: Spyros Matskoukas, Najim Dehak, Luciana Ferrer, Hynek Heřmanský, Bing Zhang and other great people I met during my work on the IARPA BEST and DARPA RATS projects.

Special thanks belongs to my parents for their support and help throughout all of my studies.

Finally, a very special thanks to Katka, my partner in life, whose support defies description and to whom I devote this work.

Contents

Nomenclature

ANN	Artificial Neural Network
ASR	Automatic Speech Recognition
BEST	IARPA Biometrics Exploitation & Science Technology
DARPA	Defense Advanced Research Projects Agency
DCF	Detection Cost Function
DCT	Discrete Cosine Transform
DET	Detection Error Tradeoff
DFT	Discrete Fourier Transform
DPLDA	Discriminative PLDA
EER	Equal Error Rate
EM	Expectation-Maximization
FPD-PLDA	Full Posterior Distribution PLDA
GMM	Gaussian Mixture Model
GSM	Global System for Mobile Communications
HMM	Hidden Markov Model
HVAC	Heating Ventilation and Air-conditioning
IARPA	Intelligence Advanced Research Projects Activity
JFA	Joint Factor Analysis
LDC	Linguistic Data Consortium
MAP	Maximum A Posteriori
MFCC	Mel-Filterbank Cepstral Coefficients
ML	Maximum-Likelihood

NN	Neural Network
PDF	Probability Density Function
PLDA	Probabilistic Linear Discriminant Analysis
PRISM	Promoting Robustness in Speaker Modeling
RATS	Robust Automatic Transription of Speech
ROC	Receiver Operating Characteristics
SAD	Speech Activity Detection
SNR	Signal to Noise Ratio
SRE	Speaker Recognition
SVM	Support Vector Machine
UBM	Universal Background Model
VAD	Voice Activity Detection

Chapter 1

Introduction

Automatic speaker recognition (SRE) is a process of comparing bio-metric signals produced by the human vocal tract and answering the question to whom the given signal belongs or simply whether two signals were produced by the same individual.

Similarly to the DNA, image of the iris, contour lines of the fingerprints, etc. — voice is a common type of bio-metric data, which every individual can produce and which is easy to capture. Thanks to its nature of being easily obtained, the the bio-metric systems based on voice find a broad use in law-enforcement and intelligence. This property, however, is not desired in the authentication systems. Therefore, in such scenarios, the voice verification is usually combined with other methods like knowing a secret password or providing additional bio-metric signals. If the voice is to be a single source of bio-metric data and the system knows the supposedly secret content of the speech and is able to use this knowledge, then we consider the SRE system as *text-dependent*, otherwise we talk about a *text-independent* system.

Speech is a very complex signal carrying not only the desired content, but also other various information. After it is produced by a vocal tract (which is characteristic to every speaker and therefore it generates most of the speaker-related information in the signal) it passes through some environment to a point where it is recorded. This environment or *channel* has a great effect on the quality of such signal, which causes the degradation in performance of SRE systems. This behavior is, of course, an important topic for research and we will address it in this work as well.

An SRE system is built with the assumption that the information relevant to the speaker in the given recording is independent on the information related to channel, language, content (in case of the text-independent system), etc. Current state-of-the-art systems are designed to decouple the information contained in the signal into the speaker- and channel-related parts. As already mentioned, the problem can be viewed as answering two possible questions: (i) Who is speaking in this recording? — then we talk about the *speaker identification* or (ii-a) Is it the same speaker speaking in these two (or even more) recordings? or (ii-b) Is this speaker speaking in this recording? — then we talk about *speaker verification*.

Both questions (ii-a) and (ii-b) represent a so-called speaker verification *trial*. If the correct answer is “yes” then the trial is called a *target trial*. If “no” is the correct answer, then we talk about a *non-target trial*.

As we can see, speaker verification constitutes a two-class problem, where the task is to decide whether a test utterance belongs to a given speaker, or, equivalently, whether a set of recordings (e.g. one enrollment and one test utterance) belongs to the same speaker.

An example for the verification task can be a scenario widely used by a law enforcement. Given some utterances belonging to a particular person, the goal is to search in a collection of data and find the recordings corresponding to the given person. A speaker verification can be turned into identification, by restricting the set of compared utterances.

Speaker identification is then a multi-class classification problem, where the task is to assign a correct label to the utterance, where each label corresponds to one of the speakers from the set of known speakers. The assumption, whether the test segment belongs to the set of known speakers, constitutes two classification problems: the *closed set identification* — the segment is always assumed to belong to one of the speakers, and the *open-set identification* — the segment does not have to belong to any of the speakers. The open-set problem is a more difficult scenario. If a new speaker is to be added to the known speaker set, a procedure called enrollment is carried out. It consists of collecting a sufficient amount of speech data, assigning it a unique speaker label and creating a corresponding *speaker model.*

1.1 Designing a Speaker Recognition System

In order to build an automatic speaker recognition system, it is necessary to transform the continuous speech in such a way that it can be used by a computer. This process consists of sampling and quantization and the result is a discrete version of the signal. According to the Shannon theorem and a property of human hearing, the sampling frequency is typically 8 kHz or more.

After the discrete signal is obtained, its further parametrization has to be performed accordingly to the type of information, which should be extracted from the signal. In speaker recognition [Reynolds, 2002], we can consider several layers of information, which can be extracted from the signal. Going from the lowest (acoustic) and the most information-rich layer to the top, we can consider the following structure:

- **acoustic**: spectral representation of the speech conveying vocal tract information as well as the content of the speech itself

- **prosodic**: features encoding the prosody (pitch, energy, syllable lengths, pauses, etc.) (see e.g. [Adami et al., 2003, Dehak et al., 2007], for a thorough overview, see [Kockmann, 2012], Sec. 1.3.2)

- **phonetic**: analysis of sequences of phonemes specific to the speaker (see e.g. [Navrátil et al., 2003])

- **idiolect**: analysis of sequences of words or short phrases [Doddington, 2001]

- **linguistic**: analysis of linguistic patters characteristic to the speaker's conversation style

Going from the lowest layer to the top, usually more training data is required to obtain enough speaker-related information for the automatic system to perform a judgment with requested confidence. This can be also demonstrated by a degradation in performance, when the systems based on the presented types of information are compared on the same amount of data [Reynolds, 2002]. Still, the information obtained by focusing on different aspects of speech is often complementary and the individual systems exploiting different levels of information can be successfully combined into a single system.

In the field of automatic speaker recognition, the use of systems built on top of the spectral features is by far the most common. The systems presented in this work are also based on this type of parametrization.

1.1.1 Feature Extraction

Methods for extracting the spectral representation of acoustic signal are based on the assumption that the signal can be considered stationary within short segments (*frames*) of typical duration in order of miliseconds (usually 10 ms). Cutting the signal into such pieces could be achieved by *windowing* the signal with a rectangular window function. However, the sharp cuts at the borders of the window would introduce high-frequency distortion in the spectrum. For this reason, a window function which attenuates the signal near its borders is used — typically the bell-shaped *Hamming-window* function [Young et al., 2006]. Considering the fact that the information in the tails of the window function is suppressed, the window length is extended (usually to 20–25 ms) and then applied with a constant shift corresponding to the intended frame-rate (again usually 10 ms). A pre-emphasis filter can be applied before actual windowing to amplify higher frequencies. The motivations for pre-emphasis are the psycho-acoustic findings about sensitivity of human hearing to different frequencies [Moore, 2012].

After the actual windowing, the power magnitude Fourier spectrum is computed for every frame, which is further parametrized into the low-dimensional representation called *feature vector*. In this work, we use the Mel-Filterbank Cepstral Coefficients (MFCC) for all presented systems.

Mel-Filterbank Cepstral Coefficients

Mel-filterbank Cepstral Coefficients have been originally introduced for Automatic Speech Recognition (ASR) [Rabiner and Juang, 1993, Davis and Mermelstein, 1980] and since then gained popularity in all fields of speech processing. For SRE, MFCCs have become a standard method of parametrization and they usually serve as a baseline for any newly proposed feature extraction method. Figure 1.1 shows the extraction steps of the MFCC feature vector for a single frame of signal and Figure 1.2 is a visualization of the frame after it is processed in different stages[1]. First of all, the absolute value of the short-term Discrete Fourier Transform (DFT) is used to extract the amplitude of the spectrum from individual frames. Then, Mel-filterbank [Rabiner and Juang, 1993] is applied to smooth the spectrum. Mel-filterbank is a set of triangular band-limited weighting functions equidistantly distributed over the *Mel scale* [Stevens et al., 1937], designed

[1] Figures 1.1 and 1.2 have been reproduced from [Burget, 2004] with kind permission of the author.

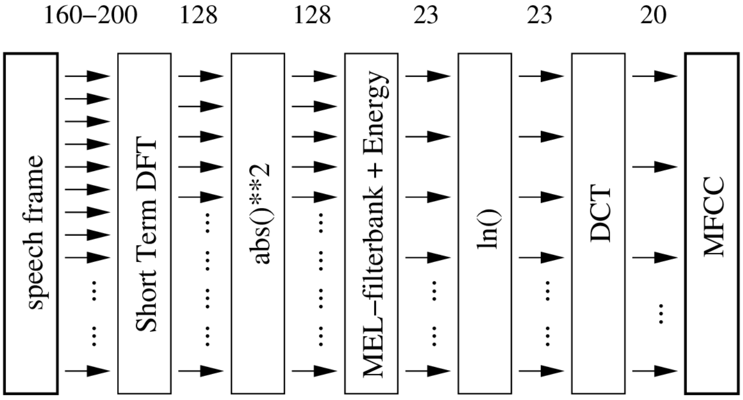


Figure 1.1: MFCC extraction steps — the numbers above the blocks show dimensionalities for frame lengths of 20 and 25 ms at sampling frequency $f_s = 8000\,\mathrm{Hz}$.

according to the properties of human hearing to provide better resolution in the lower frequencies (see Figure 1.2(c)). Given the filter bank, a vector of band energies is computed as a weighted sum of squared values of the amplitude spectrum. An overall frame energy (usually added as the zero-th coefficient) is computed as an average of squared samples. Then a logarithm of the energies is taken in agreement with the human perception of the sound loudness. Finally, the feature vector is de-correlated and its dimensionality is reduced by projection into a certain amount of Discrete Cosine Transform (DCT) bases.

Feature Derivatives

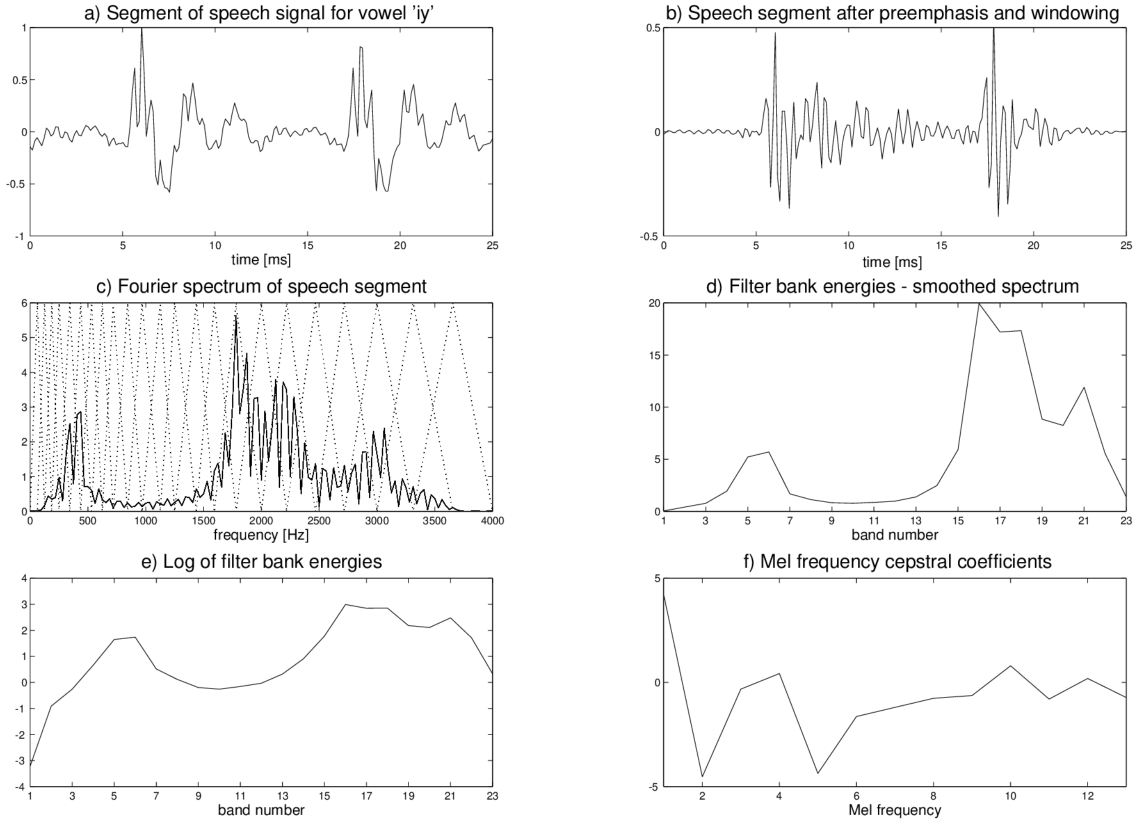
To add a dynamic information to the static cepstral features, the consecutive feature vectors are extended with their first, second, and/or third order derivative approximations. These derivatives are referred to as delta, double-delta (or acceleration), and triple-delta coefficients [Furui, 1986]. State-of-the-art SRE systems usually include first and second order derivatives. The first order derivative for a feature vector $\mathbf{c}$ in frame k is computed as a linear combination of the $\pm N$ surrounding feature vectors, i.e.:

$$\Delta\mathbf{c}(k) = \sum_{j=-N}^{N} j\,\mathbf{c}(k-j), \tag{1.1}$$

where N is in our case set to 2. Higher-order derivatives can be obtained by recursively applying the above formula to the lower-order derivatives.

Mean and Variance Normalization

It can be observed that convolutive noise will shift the means of the MFCC coefficients, while the additive noise will shrink their variance. To cope with this unwanted effect, which would cause the dynamics of individual recordings to vary, a simple mean and variance normalization [Boll, 1979, Openshaw and Masan, 1994] is performed on the whole utterance with the assumption that the channel effect is constant over the entire utterance. Especially in SRE and in the real-time scenarios, this normalization is performed


a) Segment of speech signal for vowel 'iy'

b) Speech segment after preemphasis and windowing

c) Fourier spectrum of speech segment

d) Filter bank energies - smoothed spectrum

e) Log of filter bank energies

f) Mel frequency cepstral coefficients

Figure 1.2: MFCC extraction—visualization of MFCC extraction steps for a single frame.

locally on a sliding window of 3–5 s duration. The feature vector being normalized is then in a center of the sliding window. We call this variant a *short-time* mean and variance normalization.

The normalization is computed as follows: for a k-th frame in utterance $\mathcal{X}$, the normalized i-th coefficient $\hat{c}_{\mathcal{X},i}(k)$ is computed as

$$\hat{c}_{\mathcal{X},i}(k) = \frac{c_{\mathcal{X},i}(k) - \mu_{\mathcal{X},i}}{\sigma_{\mathcal{X},i}}, \tag{1.2}$$

where the normalization parameters mean $\mu_{\mathcal{X},i}$ and standard deviation $\sigma_{\mathcal{X},i}$ are estimated on a given utterance $\mathcal{X}$.

1.1.2 Voice Activity Detection

Voice Activity Detection (VAD), also known as Speech Activity Detection (SAD) is an important pre-processing step in most speech-processing and telecommunication applications. Its purpose is to select only those frames from the analyzed utterance, which contain speech.

There are various approaches how to detect speech. We can consider the simple energy thresholding, Gaussian Mixture Model (GMM) classifier or Neural Networks (NN) trained to discriminate between speech and the rest of the audio signal.

In this work, VAD is based on a hybrid of Artificial Neural Networks (ANN) and Hidden Markov Model (HMM). It is used as a phoneme recognizer trained on the SPEECH-DAT Hungarian database [Matějka et al., 2006]. The outputs of such recognizer are strings of phonemes, of which only those corresponding to speech are used to define speech frames. The rest (all models of silence) is used to define the other class (non-speech). More details about the used VAD will be given in Section 10.1.1.

1.2 Obtaining the Speaker Verification Scores

The verification score is usually obtained by evaluating statistical models as a log-likelihood ratio between two hypotheses. The two hypotheses correspond to answers "yes" or "no" to the verification question (either (ii-a) or (ii-b) in Introduction). However, in some cases, a score can be obtained, by using a simple metric based on a distance between feature vectors characterizing the whole utterance, without explicitly training an SRE system (see Chapter 5).

Essentially, there are two approaches to the modeling: *generative* and *discriminative*. The generative approach aims at estimating the underlying distribution of the data, from which the input features can be generated. A common method to train a generative model is fitting its parameters to maximize the data likelihood. It is widely used for its simplicity and robustness. One of the advantages is its ability to be adapted if sufficient amount of new data is available. Also, some basic (but powerful) generative models can be trained without any labels and provide such representation of data that can be directly used for obtaining the verification score (see Section 5.3). Most of the models presented in this work are generative and will be described in chapters 4 to 7.

Discriminative models, on the other hand, are trained to directly predict classes from the observed data. In contrast to the generative model, given the data with corresponding labels, parameters of these models are trained to define a separation boundary between the classes.

In SRE, the discriminative training has originally been proposed in [Campbell, 2002, Campbell et al., 2006], where Support Vector Machines [Vapnik, 1995] were trained in a one-versus-many fashion, to create a model for each speaker against a cohort of impostors. This can by seen as an *asymmetric* verification approach (corresponding to the question (ii-b) in Introduction). In this work (see Chapter 9), we will build a verification system as a single classifier trained to decide whether both utterances from a given pair correspond to the same speaker (problem (ii-a) in introduction). We call this scenario a *symmetric* verification approach.

1.2.1 Score Normalization

Score normalization techniques have become important to cope with the effects of unwanted variability associated with log-likelihood ratio scores [Auckenthaler et al., 2000]. The causes of the variability can be changes in the channel as well as the intra-speaker variability that may occur across multiple sessions. Having these problems in the scores usually points to a limited capability of the models to compensate for these unwanted effects.

Usually, normalization is a linear operation which consists in a global shift and scale. The scale and shift are estimated using a separate normalization set which usually consists only of impostor speakers (speakers different than those, whose scores are being normalized). In general, normalization with shift μ and scale σ is performed as

$$s_{\text{norm}} = \frac{s - \mu}{\sigma}. \tag{1.3}$$

In this work, we apply the score normalization for systems performing asymmetric verification. Although we give the following description of different normalization techniques for the asymmetric approach, it can be easily applied to the symmetric one as well. However, current state-of-the-art techniques for the text-independent speaker verification usually do not require normalization.

Zero Normalization – Z-norm

The Z-norm is generally considered to be a means for compensating with respect to inter-speaker variability in the scores. It compensates for the biases and scales in the enrollment model scores evaluated against the test data.

The normalization constants for speaker model $\mathcal{M}$ are estimated from scores obtained by scoring a set of impostor recordings against the enrolled model and applied according to equation (1.3). Empirically, we know that these scores follow roughly Gaussian distribution. Normalizing them to zero mean and unit variance allows us to use a global speaker-independent verification threshold. Mathematically, we enforce that

$$p\left(\frac{s_{\text{imp}} - \mu_{\mathcal{M}}}{\sigma_{\mathcal{M}}}\middle|\mathcal{M}\right) \approx \mathcal{N}\left(\frac{s_{\text{imp}} - \mu_{\mathcal{M}}}{\sigma_{\mathcal{M}}}; 0, 1\right). \tag{1.4}$$

Figure 1.3 depicts this procedure as STEP 1.[2] The advantage of Z-norm is the possibility to pre-compute the normalization statistics offline.

Test Normalization – T-norm

In contrast to Z-norm, it is generally assumed that the T-norm compensates for inter-session variability between the tested utterance and a set of speaker models. The normalization constants have to be estimated online when scoring a test utterance $\mathcal{X}$. This utterance is scored against a set of impostor models and from the resulting scores, shift and scale are estimated. Again, we enforce that

$$p\left(\frac{s_{\text{imp}} - \mu_{\mathcal{X}}}{\sigma_{\mathcal{X}}}\middle|\mathcal{X}\right) \approx \mathcal{N}\left(\frac{s_{\text{imp}} - \mu_{\mathcal{X}}}{\sigma_{\mathcal{X}}}; 0, 1\right). \tag{1.5}$$

The method is marked as STEP 3 in Figure 1.3.

[2] The picture was reproduced from [Glembek, 2012] with kind permission of the author.

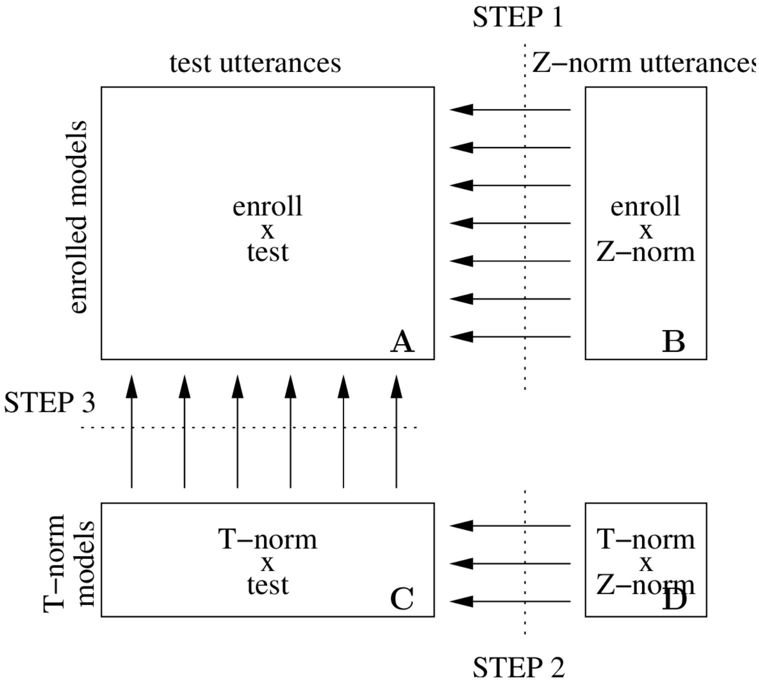


Figure 1.3: Application of ZT-norm. The boxes denote matrices of complete scores, i.e. all models against all scored utterances. Steps 1 and 3, if applied separately correspond to the Z-norm and T-norm, respectively.

ZT-norm

ZT-norm is a combination of both above introduced techniques. It is equivalent to subsequently performing Z- and T-norm. The procedure consists of performing the steps 1–3 as shown in Figure 1.3: First, the Z-norm is applied to the scores of enrolled models and T-norm models against the test utterances (steps 1 and 2). Next, T-norm parameters are estimated on Z-normalized T-norm scores. Finally in step 3, these parameters are applied to (again Z-normalized) test scores.

S-norm

S-norm is a technique, which takes advantage of the systems designed according to the symmetric scenario. It can achieve a similar effect to that of ZT-norm, while performing less computations. Usually, a single held out cohort of speakers serves as a Z-norm cohort as well as segments for training T-norm models. Z-norm and T-norm is independently applied to obtain two sets of normalized scores. Final scores are simply obtained by averaging the corresponding scores from these two sets.

1.3 Motivation and Contribution

My work on the topics of this thesis started when I was building subsystems for the NIST SRE 2010 in the team of people from Agnitio, Brno and CRIM (ABC). Later, during the 2010 BOSARIS workshop held in Brno, I was working on the analysis of systems submitted by the ABC team to the NIST SRE 2010. The main focus was on Probabilistic Linear Discriminant Analysis using i–vectors as features as it showed excellent results in the evaluations. At that time, it was already becoming apparent that PLDA and i–vectors will become a new state-of-the-art in SRE. I was also working with Lukáš Burget on one of the research directions, where the goal was to formulate a discriminative way of training the PLDA-like model. The goal of obtaining a discriminatively trained SRE system based on the PLDA was successfully achieved [Burget et al., 2011, Cumani et al., 2011] and for a short time (until the introduction of the i–vector length normalization [Garcia-Romero, 2011]), this technique was providing the best results. I continued my work on discriminative training, dataset design and calibration [Ferrer et al., 2012, Ferrer et al., 2011b] as a member of BUT and SRI team in the IARPA Biometrics Exploitation & Science Technology (BEST) program. Later on, I was working with Sandro Cumani on various topics in SRE, the main being the extension of the PLDA model [Cumani et al., 2014], which takes into account the uncertainty about the i–vector. As the uncertainty of the i–vector estimate depends mainly on the duration of speech segments from which the i–vectors are extracted, the proposed extension turned out to be effective mainly for short segments. At the same time when developing the PLDA extension, I was also working both on a speaker- and language modeling, calibration and fusion [Plchot et al., 2013] for a DARPA RATS (Robust Automatic Transription of Speech) project in a team led by BBN Technologies. Working on RATS allowed me to compare generative PLDA with its discriminative counterpart in a very noisy and degraded acoustic environment.

1.3.1 Claims

The goal of this thesis is to investigate the contemporary state-of-the-art techniques in text-independent speaker verification field. The main focus is on the analysis and further improvement of the Probabilistic Linear Discriminant Analysis (PLDA). The main contributions can be summarized in the following points:

- **Analysis of PLDA**: I analyzed the performance of presented methods on various datasets representing different levels of acoustic signal distortions and channel variabilities. Also, a direct comparison of the main techniques considered as the state-of-the-art before introduction of PLDA is provided on a common dataset.

- **Extension of PLDA**: The proposed extended PLDA model takes into account an uncertainty of the input features, which improves performance on short speech segments with respect to the original PLDA model.

- **Discriminative training of PLDA**: The proposed discriminative approach to the PLDA model training offers an interesting alternative to the currently preferred generative approach. Presented results suggest that the discriminatively trained PLDA model offers well calibrated outputs and therefore poses as a viable option for a practical use.

1.3.2 Structure of the Thesis

The thesis is organized as follows:

- **Chapter 2** introduces the composition and design of various datasets that are used throughout this work.
- **Chapter 3** describes the evaluation metrics, which are used to measure the performance of SRE systems presented in this work.
- **Chapter 4** outlines the basics of acoustic modeling and subspace methods. It provides theoretical prerequisites for speaker modeling techniques described by this thesis.
- **Chapter 5** outlines the concept of i–vectors as features for speaker modeling.
- **Chapter 6** presents the concept of the Probabilistic Linear Discriminant Analysis, which serves as a basic model for the proposed techniques.
- **Chapter 7** presents the extension of the Probabilistic Linear Discriminant Analysis model, taking into account the uncertainty of the i–vector generation process.
- **Chapter 8** discusses the problematic of i–vector normalization and presents its application to the extended PLDA model.
- **Chapter 9** presents a discriminative approach to the training of the PLDA model.
- **Chapter 10** provides results and comparison of SRE systems based on the presented techniques. The results are presented on various datasets representing different acoustic conditions.
- **Chapter 11** concludes this work.

Chapter 2

SRE Databases and Various Evaluation Tasks

Various datasets were used throughout this work, which allowed us to explore the behavior and performance of the speaker recognition system under different acoustic conditions. The acoustic conditions are specific to the source of the data and usually represent a common factor affecting the data. These factors contribute to an unwanted variability with respect to discriminating between speakers — we want to remove it or take it into account in our systems. The most common source of unwanted variability in the data is the acoustic channel, through which the audio was either recorded or transmitted. It can be a telephone line, various types of microphones used during recording, use of a specific codec or a compression during transmission, or even distortions caused by using specific equipment in the Radio-Link transmission.

On top of the effects of the transmission channel, we can often encounter other additive or convolutive noise in the data. The most common sources of such noise are cross-talks from the other side of the two-way conversation, reverberation, presence of the ambient noise (i.e. HVAC[1] noise) or background noise in the environment (i.e. common babble noise in cafeteria).

In some situations, especially if we are designing a general system, we can expect to encounter recordings coming from people of different nationalities talking in various situations. Therefore we are dealing with variability caused by different languages, speaking style or age of the speakers.

In the following sections, we will describe sources of the datasets, which were used either directly or which we modified or re-arranged in order to simulate and evaluate the performance of speaker recognition systems in the diverse acoustic conditions.

2.1 NIST

Databases created for the purposes of the NIST SRE evaluations represent a key element in SRE-related research and recent NIST evaluations serve as a common benchmark and

[1] HVAC stands for Heating Ventilation and Air-conditioning. It is a very common type of noise, which led NIST to design a special noisy condition with such type of noise in NIST SRE 2012 evaluations.

a base for designing training datasets for the SRE community. Each evaluation consists of different common evaluation conditions — subsets of trials in the core test that satisfy additional constraints. The constraints are usually designed to group the data according to some characteristic such as channel, nominal length, number of utterances per trial side, speaking style, etc.

Most of the experimental results are reported on the selected conditions of official NIST Speaker Recognition Evaluation tasks. Especially the extended conditions defined for the NIST SRE 2010 [NIST, 2010] are still widely used benchmark for most state-of-the-art general SRE systems. The importance of the data released for the previous evaluations has not weakened as these are continuously being moved into the datasets used for system training and development.

2.1.1 NIST SRE 2012

The NIST SRE 2012 [NIST, 2012] were different than previous NIST SRE's. In all previous evaluations, the evaluation set contained both the enrollment and the test data newly collected from unseen speakers. In SRE 2012, however, most of the target speakers were taken from previous SRE corpora. Furthermore, the participants were forming the enrollment part of the trials themselves by using data released for previous evaluations. This has resulted in having often tens of segments in the enrollment, which was not, by far, the case before.

Similarly to the NIST SRE 2010, all speech is expected to be in English, though English may not be the first language of some of the speakers. The acoustic channels are represented by typical interviews recorded over various types of microphones and telephone calls established using various handsets. To make the data more challenging and different from previous evaluations, NIST had corrupted some of the test segments with additive noise. The noise was represented by various samples of HVAC and crowd babble noise. It was mixed with the original segments at various SNRs. Also, part of the telephone recordings was collected in a naturally noisy environment. The selection of the noisy environment was left on the decision of the person making a phone call.

Some of the experimental results will be reported on the extended conditions of the core NIST SRE 2012 task. There are five common conditions with the following characteristics [NIST, 2012]:

1. All trials involving multiple segment training and interview speech in test without added noise in test;

2. All trials involving multiple segment training and phone call speech in test without added noise in test;

3. All trials involving multiple segment training and interview speech with added noise in test;

4. All trials involving multiple segment training and phone call speech with added noise in test;

5. All trials involving multiple segment training and phone call speech intentionally collected in a noisy environment in test.

Similarly to the previous evaluations, NIST does not evaluate on non-target trials formed as different-sex trials, so in theory the trial-set can be divided into female and male subset. However, unlike with the NIST SRE 2010, we will report results on the full set of trials. A short summary of the conditions is given in Table 2.1.

Table 2.1: Training and test conditions of the NIST SRE 2012 evaluation. In the second column, known and unknown denotes numbers of non-target trials formed from previously released data and from the new data, respectively.

Condition	targets	known	unknown	Channel
1	3860	10985377	11349426	interview, no added noise
2	7354	10312118	2088834	phone call, no added noise
3	5127	12444672	4804500	interview, added noise
4	7176	9471219	124830	phone call, added noise
5	3883	5119130	77745	phone call from a noisy environment

2.1.2 NIST SRE 2010

The NIST SRE 2010 evaluations still constitute a favorite benchmark thanks to its similarity with all previous evaluations and relative simplicity of designing the development set for this task.

As usual, the new data were introduced to the community in SRE 2010. In addition to the microphone channels already present in SRE 2008, seven new microphone channels were added in 2010. The new sources of intrinsic variability were various levels of speakers' vocal effort and conversations from speakers who participated in older speaker collections (so called graybeard data). Also, the segments of variable lengths were added to the test. Similarly to SRE 2012, the data are in English, however, English does not have to be the native language of all speakers.

This evaluation is different from the previous SRE's, not only because of increasing amount of evaluation data, but mainly because of the new primary metric. Compared to the earlier SRE's, the detection cost function (DCF, see Section 3.1) serving as a primary metric was modified. The new metric was designed to penalize false alarms more severely, which was achieved by decreasing the cost of miss and the target trial probability. In comparison to the NIST SRE 2008, where the cost ratio of the false-alarm to miss was 10:1, the new metric increased it 100 times to 1000:1. The two metrics of SRE 2008 and 2010 are often referred to as “old” and “new” DCF. We will describe these metrics in detail later.

The new operating point defined by the modified metric, however, has brought another challenge. In order to obtain statistically significant results at low false-alarm rates, the number of trials had to be substantially increased. This was the reason for releasing an extended set of trials short after the evaluations. The number of trials in the extended set was nearly 6.5 million — an order of magnitude more than in the original core set.

Most of the experimental results in Chapter 10 are reported on the extended conditions of the core NIST SRE 2010 task. The NIST defined nine common conditions with the following characteristics [NIST, 2010]:

1. All trials involving interview speech from the same microphone in training and test,
2. All trials involving interview speech from different microphones in training and test,
3. All trials involving interview training speech and normal vocal effort conversational telephone test speech,
4. All trials involving interview training speech and normal vocal effort conversational telephone test speech recorded over a room microphone channel,
5. All different telephone number trials involving normal vocal effort conversational telephone speech in training and high vocal effort conversational telephone speech in test,
6. All telephone channel trials involving normal vocal effort conversational telephone speech in training and high vocal effort conversational telephone speech in test,
7. All room microphone channel trials involving normal vocal effort conversational telephone speech in training and high vocal effort conversational telephone speech in test,
8. All telephone channel trials involving normal vocal effort conversational telephone speech in training and low vocal effort conversational telephone speech in test,
9. All room microphone channel trials involving normal vocal effort conversational telephone speech in training and low vocal effort conversational telephone speech in test.

Again, NIST defines only same-sex trials and we will report results mainly on the female subset of all trials. The split between male and female trial subset is very often used in the publications and the female subset is generally selected because it is slightly harder. Also, we will concentrate on the first five conditions from the list above. The short summary of the conditions is given in Table 2.2.

Table 2.2: Training and test conditions of the NIST SRE 2010. In the second and third column, “tar” and “non” denote numbers of target and non-target trials, respectively.

Condition	Female		Male		Training	Test
	tar	non	tar	non		
1	2326	449138	1978	346857	interview	interview
2	8152	157394	6932	121558	interview	interview
3	1958	334438	2031	303412	interview	telephone
4	1751	392467	1886	364308	interview	microphone
5	3704	233077	3465	175873	telephone	telephone

2.1.3 NIST SRE 2004–2008

In this work, all of the data from years 2004–2008 NIST SREs were included into the background data set and used for training. The databases for these evaluations contain various languages with the dominance of English.

The NIST SRE 2008 [NIST, 2008] was significant by including interview speech into the evaluation. The interviews were recorded over several different microphones, which allowed an exploration of the effects caused by this kind of channel variability.

The NIST SRE 2005 [NIST, 2005] and 2006 [NIST, 2006] corpora consist of telephone call conversations recorded over land-line as well as cellular phones. Additionally, telephone calls were recorded over auxiliary microphones of different kinds. More detailed description of these datasets can be found in [Glembek, 2012] and in corresponding evaluation plans.

2.2 Other Datasets

To design a robust speaker recognition system with top performance, the amount of data plays a crucial role. Especially, the amount of different speakers needs to be large in order to model inter-speaker variability. Within the speaker recognition community, the Switchboard and Fisher databases released by Linguistic Data Consortium (LDC) are widely used as a part of training data and it is also the case in our systems. The detailed description of the two databases was taken from [Glembek, 2012, Kockmann, 2012] with the kind permission of the authors.

2.2.1 Switchboard

Switchboard 2 Phase II [Graff et al., 1999] was released in 1999 and consists of 4,472 five-minute telephone conversations involving 679 participants which were mainly recruited from US college campuses. Each speaker participated in at least 10 calls. Switchboard 2 Phase III [Graff et al., 2002] had been recorded between 1997 and 1998 in the American South and consists of 2,728 calls from 640 participants (292 Male, 348 Female) which are all native English speakers. Both of these corpora only consist of land-line calls. Switchboard Cellular Part 1 [Graff et al., 2001] was recorded until 2000 and mainly focuses on cellular phone technology. It consists of 1,309 calls, or 2,618 sides (1,957 GSM), from 254 participants (129 Male, 125 Female), under varied environmental conditions. Switchboard Cellular Part 2 [Graff et al., 2004] was released in 2004 and consists of 2,020 calls, or 4,040 sides (2,950 cellular, 2,405 female, 1,635 male), from 419 participants.

2.2.2 Fisher English

Fisher English is a collection of conversational telephone speech collected in 2003 by LDC. The database protocol was created at LDC to address a critical need of developers trying to build robust Automatic Speech Recognition (ASR) systems. A very large number of participants make a few calls of short duration speaking to other participants (in English), whom they typically do not know, about assigned topics. This maximizes inter-speaker variation and vocabulary breadth, although it also increases formality. The database contains 11,699 recorded telephone conversations, each lasting up to 10 minutes.

2.3 PRISM

The last years have seen a dramatic increase in the amount of data to be processed. The new data brought new and extended existing types of channel variabilities such as: different speaking styles, use of a wide variety of microphones, as well as the speech recorded with different levels of vocal effort or recorded in the noisy environment.

During our efforts to build the general-purpose and robust SRE systems for the IARPA BEST program, we have developed a dataset focused on addressing different variabilities and distortions present in the speech data. We have designed the database on top of publicly available data and made it publicly available for the research community [Ferrer et al., 2011b, Ferrer et al., 2011a].

The PRISM (Promoting Robustness in Speaker Modeling) evaluation set is a large speaker recognition set based on NIST SRE data released from 2005 to 2010, where the scope is extended to additional types of variabilities, namely noise and reverberation. In addition, it includes variabilities already seen in one or more NIST SREs, namely language, channel type, speech style and vocal effort level.

2.4 Data Description

The PRISM evaluation set is created using data from all NIST SREs beginning with the year 2005 (that is, SREs 2005, 2006, 2008 and 2010). NIST SRE 2004 data, along with Fisher and Switchboard data, are also included in the database, although used only for training purposes, not to create evaluation trials.

The evaluation set is divided into different subsets (conditions) designed to test the effect of various kinds of variability: language, noise, reverberation, speech style, channel, and vocal effort. Additionally, the NIST SRE 2010 conditions for 1–side and 8–side training are included as separate sets for the ease of comparison with previous results.

Only segments from the SRE databases of lengths that were included in the core conditions in the corresponding evaluations are used to create trials.

The "language" condition leverages data from multiple corpora released for NIST SRE evaluations to assess speaker recognition performance under multiple languages, including same-language and cross-language trials. The "reverb" and "noise" conditions are created from a clean data set that is artificially degraded at different signal-to-noise ratio (SNR) levels, using different real noises, and different reverberation delays and room types. These simulated sets are carefully crafted so that audio files and tools used to simulate the corresponding degradations are all openly available and at no cost. The other conditions use data from NIST SRE 2008 and 2010 to address the effect of channel type, speech style and vocal effort level.

Detailed description of the individual evaluation conditions as well as the reference results with a baseline system at the time of creation of the dataset can be found in [Ferrer et al., 2011b]. It is out of the scope of this work to discuss the detailed composition of the PRISM set, but we shall at least mention how the noisy and reverberated conditions were created as the process demanded us to modify the original data released by NIST.

2.4.1 Creating Noisy and Reverberated Sets

The "noise" and "reverb" sets are created by adding real noise (i.e., recorded noise samples) and reverberation to data extracted from the NIST SRE 2010 and 2008 corpora. To limit the influence of other than noisy and reverberated channel, only clean microphone data is selected from those corpora. Specifically, microphone 2 (lavalier microphones) segments are chosen from both interview and telephone conversations.

In case of "noise", we selected 15 cocktail noise samples from the free sound repository Freesound.org [Freesound, 2010]. These noise samples were collected in bars, cafeterias, offices, and airports. The samples were inspected to remove single-speaker foreground speech sounds and artifacts (e.g., clicks). Afterwards, these 15 noise samples were mixed to the clean segments at 20, 15, and 8 dB SNRs, using the publicly available filtering and Noise adding Tool – FaNT [Hirsch, 2005]. To avoid the optimistic scenarios of matched noise environments, different noises are added to training, enrollment, and test samples.

Reverberation is added to the clean signals using different reverberation times (RT) of 0.3, 0.5 and 0.7 second. Initially, a set of candidate rooms were generated using the rir tool [McGovern, 2004], which allows for the modeling of a room impulse response for parameters of room size, microphone and speaker location, wall, floor and ceiling reflection coefficients, speed of sound, and so on. The rooms were modeled so as to cover common configurations of size, reflectivity, and source and microphone locations and only those configurations resulting in RTs close to 0.3, 0.5 or 0.7 were used. In total, twelve rooms were modeled for training (four for each RT), three for test, and three for enrollment (one for each RT in each case). Finally, the "fconv" tool from the same toolkit was used to generate the reverberated signals by convolving the room impulse responses with the audio files.

2.4.2 Selected Conditions

To analyze how systems can deal with different inter-session variabilities, we chose a representative set of PRISM conditions for reporting results. The set of conditions is the same as already defined in [Ferrer et al., 2012]. Here, we will list the names of the conditions with a short description. For the extensive description, we refer the reader to [Ferrer et al., 2012, Ferrer et al., 2011b].

telp English telephone calls over telephone channel for both signals in the trial.

tela English telephone calls over either telephone or microphone channels for both signals in the trial.

int English interviews over microphone channels for both signals in the trial.

vel Normal vocal effort English conversations versus normal, low and high vocal effort English conversations.

lan Trials where both signals are telephone conversations in the same language, which can be either English, Chinese, Russian, Arabic or Thai.

noi Clean and noisy microphone interview signals with different SNR levels tested against each other.

rev Clean and reverberated microphone interview signals with different RTs tested against each other.

2.5 DARPA RATS

The Robust Automatic Transcription of Speech (RATS) is a DARPA-sponsored program, with the goal of creating technology capable of accurately determining speech activity regions, detecting key words, and identifying language and speakers, in highly degraded, weak and/or noisy communication channels. The data sets used in RATS are obtained by retransmitting pre-existing or newly collected telephone conversations in multiple languages over various types of channels, and aim to capture/simulate the acoustic environment present in current radio-based two-way communications systems used by the law enforcement, emergency, air traffic control, etc.

By its nature, these radio means of communication are sensitive to many factors which can degrade or change the quality of the transmission. The most important are background radio interference, atmospheric conditions, used bandwidth and background additive noise. All of these factors greatly increase the unwanted channel variability present in the audio.

The channel distortions present in this type of data forced us to revisit every step in the chain of technologies which lead to the final SRE system [Plchot et al., 2013]. Most of the state-of-the-art systems are designed for much cleaner telephone conversations or interviews recorded with high-quality microphones in relatively low-noise environment compared to RATS data.

In order to create the speaker recognition systems for RATS, we had to begin by developing noise-robust models for voice activity detection based on both supervised and unsupervised methods. In addition, we experimented with various types of acoustic features in order to see their effect on the behavior of the system under noise. In this work, however, we will limit the experiments to the standard MFCC features to demonstrate the performance of proposed techniques under such degraded acoustic conditions.

2.5.1 Data Description

The Linguistic Data Consortium (LDC) provided the training and test data for the RATS participants. The audio recordings were selected from existing and new data sources as follows:

- NIST SRE 2004 (English, Arabic, Chinese, Russian, Spanish)

- RATS-LDC (Levantine Arabic, Farsi)
- RATS-Appen (Levantine Arabic, Farsi, Pashto, Dari, Urdu)
- CallFriend Farsi
- Fisher Arabic Levantine
- Fisher English
- NIST LRE[2] (various languages)

All recordings were retransmitted through 8 different noisy communication channels, labeled by the letters A through H [Walker and Strassel, 2012]. A "push-to-talk" (PTT) transmission protocol was used in all channels except G. PTT states produce some regions where multiple non-transmission (NT) segments may occur. As a result, the amount of usable audio decreases after retransmission.

It should be noted that among the data sources listed above, only the first three were annotated with speaker labels. Data from the other sources was used to train universal background models and i-vector extractors. We used the "dev" subset of the RATS-LDC and RATS-Appen corpora[3] to define speaker enrollment and test samples. The rest of the RATS-LDC and RATS-Appen data, along with the NIST SRE 2004 set was used for system training.

There is also a separate blind "progress" test set, which is used to measure year-to-year progress on the RATS SRE task. The progress set consists of speakers from the 5 target languages (Levantine Arabic, Farsi, Pashto, Dari, Urdu). Each speaker has 10 recording sessions, re-transmitted over the 8 noisy channels as described above. For each speaker, 6 of the sessions are used for enrollment and 4 for testing, randomly sampled from the noisy channels. The progress set defines multiple testing conditions, depending on the amount of speech present in enrollment and testing samples. The following test-enroll conditions are evaluated (numbers indicate nominal amount of speech in seconds): 120–120, 30–30, 30–10, 30–3, 10–10, 10–3, 3–10, 3–3. Unfortunately, at the time of writing this thesis, the reference labels for this evaluation database are not available and therefore the reported results are only on the development set, which was used for the calibration and fusion in our RATS SRE submission.

Only recordings from the 120 s condition were released for training and development. We therefore had to construct our own development samples for the shorter durations from the 120 s audio files, based on our voice activity detection to simulate the "progress" test set.

[2] Mostly the data from NIST Language Recognition Evaluation in 2009, re-transmitted through the RATS channels.

[3] LDC catalog ids: LDC2012E49, LDC2012E63, LDC2012E69.

Chapter 3

Evaluation Metrics and Criteria

As was mentioned earlier in Section 1.2, the speaker verification system is basically a two-class pattern recognizer, which is expected to classify a speaker verification trial t. In general, the verification trial can be composed of two sets of speech segments (enroll and test) and it is assumed that all segments in the individual sets belong to a single speaker. The classes to recognize represent two different hypotheses, which can be inferred from the trial: (i) the same speaker hypothesis H_s saying that both sets of segments for a given trial belong to the same speaker (often referred to as a *target trial*), or (ii) an opposite proposition of different speaker hypothesis H_d saying that the two sets of segments were uttered by two different speakers (often referred to as a *nontarget trial*).

In order to quantify the performance of such recognizer, we need a supervised set of trials $\mathcal{T}$, where each trial $t \in \mathcal{T}$ is associated with a label corresponding to a different- or same-speaker hypothesis $t \in \{-1, 1\}$. Ultimately, the goal of the verification system is to assign correct labels to the tested trials. During this process, two types of detection errors can arise: *false alarms* (FA) — different-speaker trials are incorrectly classified as same-speaker trials; and *missed detections* (Miss)[1] — when the same-speaker trials are incorrectly classified as different-speaker trials. To evaluate these error rates, it is convenient to split the set of all trials into the sets containing only same- and different-speaker trials denoted as $\mathcal{T}_s$ and $\mathcal{T}_d$, respectively. Then, for a given test set, we can estimate the miss and false alarm rate as:

$$\begin{aligned} p(\text{miss}|\mathcal{T}) &= \frac{N_{\text{miss}}}{|\mathcal{T}_s|}, \\ p(\text{fa}|\mathcal{T}) &= \frac{N_{\text{fa}}}{|\mathcal{T}_d|}, \end{aligned} \tag{3.1}$$

where $|\mathcal{T}_s|$ and $|\mathcal{T}_d|$ are the numbers of same- and different-speaker trials, respectively, and N_{fa} and N_{miss} are the numbers of false alarms and missed detections made by the system, respectively.

[1] The "FA" and "Miss" are adopted by NIST and widely used in the speaker recognition community. These names of the errors correspond to the perspective of law enforcement when the system is used to search for some suspect individual. In the field of bio-metric authentication, the terms *false accept* and *false reject* are used to reflect the other perspective.

The output of the recognizer is usually a score, which reflects the confidence of the system. Preferably, it is a calibrated log-likelihood ratio between the two hypotheses. Formally, the score for a trial t is given as

$$s = \log \frac{p(t|H_s)}{p(t|H_d)}. \tag{3.2}$$

The higher value of the score reflects a higher confidence for same-speaker hypothesis and lower value for different-speaker hypothesis. Eventually, the score is converted to a hard decision by *thresholding*. Moving the threshold τ changes the proportion of the two error rates $p(\text{miss}|\mathfrak{T}, \tau)$ and $p(\text{fa}|\mathfrak{T}, \tau)$, letting the user choose the desired *operating point* of the system. This way, the error rates also depend on the selected threshold.

These error rates are favorite and simple criteria to determine the costs of operating some particular system. Often the end user of the system is able to estimate costs related to each type of error, e.g. the time of an analyst processing false detections, or from the other perspective, the costs related to many unsuccessful authentications while accessing a banking account. For the user, it is then convenient to set the threshold in such a way that the system works with an acceptable error rate of one kind and then judge the system according to an error of the other kind. For example, a law enforcement agency would like to set an acceptable miss rate and at the same time minimize the time spent by analysts to process false alarms.

3.1 Detection Cost Function

Detection Cost Function (DCF) has been defined by NIST as a metric for evaluating the verification systems, which focuses on a particular operation point of interest. With some parameter adjustments or modifications in its definition, it serves as the primary criterion in NIST Speaker Recognition Evaluations. It is designed to consider the overall costs based on the two types of detection errors. For the evaluations prior to 2012, it is defined as a weighted sum of the false alarm probability and the miss-detection probability:

$$\text{DCF} = C_{\text{miss}}\, p(\text{miss}|\mathfrak{T}, \tau)\, p(H_s) + C_{\text{fa}}\, p(\text{fa}|\mathfrak{T}, \tau)\, p(H_d) \tag{3.3}$$

with

$$p(H_d) = 1 - p(H_s)\ , \tag{3.4}$$

where C_{miss} and C_{fa} are the relative costs of the detection errors, and $p(H_s)$ and $p(H_d)$ are the prior probabilities for the trial being same- and different-speaker, respectively. The triplet $\langle C_{\text{miss}}, C_{\text{fa}}, p(H_s)\rangle$ defines the target *operating point* corresponding to the desired application, for which the system is being evaluated. Note that the metric requires the system to make hard binary decisions as an explicit speaker detection is required for each trial. The common values, as defined by NIST for the purposes of SRE evaluations, are given in Table 3.1. Until the 2008, NIST had used values referred to as “old DCF”. In 2010, NIST introduced the new set of values, referred to as “new DCF”. The goal in 2010 was to emphasize the importance of applications operating at very low false alarm rates. The two metrics are shown as circles and squares in the DET plot which will be introduced in Section 3.2 (Figure 3.1).

Table 3.1: *Common NIST DCF parameters (applications)*

	C_{fa}	C_{miss}	$p(H_s)$
DCF_{old}	1	10	0.01
DCF_{new}	1	1	0.001

To make the measure more intuitive and to allow the comparison of difficulty of various evaluation sets, C_{Det} is further normalized by C_{Default} — the best *a-priori* cost that could be obtained without processing the input data, i.e. the one that would be obtained by accepting or rejecting all trials, whichever is smaller:

$$C_{\text{Default}} = \min \left\{ \begin{array}{l} C_{\text{miss}}\, p(H_s) \\ C_{\text{fa}}\, p(H_d) \end{array} \right. \tag{3.5}$$

and

$$C_{\text{Norm}} = C_{\text{Det}}/C_{\text{Default}}. \tag{3.6}$$

The cost is computed from the actual hard decisions and the threshold for making decisions is often set by an evaluee to minimize the cost on some development set. The metric is then referred to as the actual DCF or *act-DCF*. NIST also computes a minimum possible DCF, referred to as a *min-DCF*, by setting the optimal threshold for the given test set:

$$\begin{aligned} \min \text{DCF} = \min_{\tau}\ & [C_{\text{miss}}\, p(\text{miss}|\mathcal{T}, \tau)\, p(H_s) \\ & + C_{\text{fa}}\, p(\text{fa}|\mathcal{T}, \tau)\, p(H_d)] \,. \end{aligned} \tag{3.7}$$

The difference between the act-DCF and min-DCF is referred to as a *calibration loss*. The smaller the difference, the better the system is *calibrated*. We will further discuss the calibration in Section 3.4.

3.1.1 Analytically Setting the Threshold

If the scores are well-calibrated log-likelihood ratios (3.2), the user can set the score threshold analytically to make an optimal, cost-effective Bayes decision. The parameters of the operating point $\langle C_{\text{miss}}, C_{\text{fa}}, p(H_s)\rangle$ can be absorbed to a single *effective prior* P_{tar} defining the target application as

$$\text{logit}\, \text{P}_{\text{tar}} = \text{logit}(p(H_s)) + \log \frac{C_{\text{miss}}}{C_{\text{fa}}}. \tag{3.8}$$

where the logit function is an inverse of the logistic sigmoid and is defined as

$$\text{logit}(x) = \log \frac{x}{1-x} \tag{3.9}$$

The optimal threshold for the scores, which can be interpreted as log-likelihood ratios is then

$$\tau = -\text{logit}\, \text{P}_{\text{tar}}. \tag{3.10}$$

Having a system which provides good log-likelihood ratios and using these rules, we can treat the our system as an application independent recognizer and use it for a wide range of possible applications just by correctly setting the desired operating point.

3.1.2 Primary Metric of NIST SRE 2012

Since most of the test speakers were known prior to evaluation, NIST has altered the primary metric to reflect the situation when the two types of non-target trials are evaluated: (i) *known non-targets* coming from the previously known speakers and (ii) *unknown non-targets* coming from the unknown newly released speakers [NIST, 2012, Martin et al., 2014]. As the official scoring metric in previous SREs was simply a linear combination of the miss rate and the false alarm rate, for NIST SRE 2012 it was appropriate to consider two different false alarm rates — that of the known and unknown non-targets. For the primary core and extended task, it was decided that for the non-target trials the prior probability of a known speaker P_{known} is set to 0.5. Participants of the evaluations could also submit contrastive systems which would assume that all non-target trials would come either only from known or only from unknown speakers. The prior probability for non-target trials being formed from known speakers is then set to one or zero. Incorporating these two false alarm rates into (3.3), we get:

$$\begin{aligned} \mathrm{DCF} = {} & C_{\mathrm{miss}}\, p(\mathrm{miss}|\mathcal{T},\tau)\, p(H_s) \\ & + C_{\mathrm{fa}}\, p(H_d) \\ & \times \left(p(\mathrm{fa}_{\mathrm{known}}|\mathcal{T},\tau)\, P_{\mathrm{known}} + p(\mathrm{fa}_{\mathrm{unknown}}|\mathcal{T},\tau)\,(1 - P_{\mathrm{known}})\right). \end{aligned} \tag{3.11}$$

Additionally, to emphasize the importance of a calibration over wider range of operating points, it was decided to weight the metric for two different prior probabilities of target trials:

$$p(H_{s\_1}) = 0.01 \qquad p(H_{s\_2}) = 0.001. \tag{3.12}$$

Taking into account also this fact, we normalize as:

$$\begin{aligned} C_{\mathrm{Norm}}(\beta_k) = {} & p(\mathrm{miss}|\mathcal{T},\tau)\times \\ & \beta_k \times \left\{ \begin{array}{r} P_{\mathrm{known}}\, p(\mathrm{fa}_{\mathrm{known}}|\mathcal{T},\tau)+ \\ (1 - P_{\mathrm{known}})\, p(\mathrm{fa}_{\mathrm{unknown}}|\mathcal{T},\tau) \end{array} \right\}, \end{aligned} \tag{3.13}$$

where

$$\beta_k = \left(\frac{C_{\mathrm{fa}}}{C_{\mathrm{miss}}}\right)\left(\frac{1 - p(H_{s\_k})}{p(H_{s\_k})}\right). \tag{3.14}$$

Finally, the primary metric is an average of the cost functions for these two priors:

$$C_{\mathrm{Primary}} = \frac{C_{\mathrm{Norm}}(\beta_1) + C_{\mathrm{Norm}}(\beta_2)}{2}. \tag{3.15}$$

3.2 DET Plot

It is always desirable to produce a graphical representation of the performance to compare individual systems. In the SRE community, the Detection Error Tradeoff (DET) plot is commonly used [Martin et al., 1997] to visualize the performance over a wide range of operating points (thresholds). This plot corresponds to a min-DCF metric in the sense that a threshold optimization is performed on the whole evaluation set. Therefore similarly to the min-DCF, the DET plot is not sensitive to calibration as it depends only on the order of scores and not on their actual values. This property can be very useful for comparing systems during development as the calibration step is usually done in the very last phase and very often on the same test set (development set), which is used for the system comparison.

The DET plot is derived from the empirical Receiver Operating Characteristics (ROC) curve, which plots the detection probability as a function of false alarm probability. The DET plot is then obtained by transforming both axes of ROC plot with a non-linear probit transformation. After the transformation, the range of the x and y axes is moved from $[0, 1]$ to $[-\infty, +\infty]$. The axes x and y represent the probabilities of false alarms and miss detections, respectively. The individual operating points of an interest can be plotted on the DET curve, as well as the regions representing the statistical significance. In SRE, the region of statistical significance is often determined by the *Doddington's rule of 30*. Shortly, the rule says, that for a meaningful evaluation, one needs at least 30 false alarms and at least 30 misses. For a detailed interpretation, see appendix B of [Brümmer, 2010b] or [Doddington, 1998]. An example of a DET and ROC plot comparing two different systems denoted as PLDA and DPLDA on the PRISM "rev" test is shown in Figure 3.1.

3.3 Equal Error Rate

Equal Error Rate (EER) is a common measure characterizing the performance of a biometric system. It is defined as a location on a ROC or DET curve, where the false alarm rate and miss rate are equal. It can be shown [Brümmer, 2010b], that this point acts as a scalar summary of the whole ROC curve and it is insensitive to calibration. The value of EER gives a rough idea, how close the ROC curve is to the axes or how close is the DET curve to the origin and therefore its value can serve as a very approximate comparison between the systems. Even though the properties of this measure seem to be attractive, it is not very useful in practical applications which usually operate either in a region of low false alarm rate (e.g., authentication systems) or low miss rate (e.g., law enforcement). In Figure 3.1, the point is shown as star mark.

3.4 Normalized Bayes Error-rate Plot

So far, the quality of calibration was presented as a difference between the act-DCF and min-DCF values. This way, however, we can only see how good the calibration is in a single operating point. It is useful to plot this calibration loss over a wide range of operating points. We will use the normalized Bayes error-rate plots introduced in [Brümmer, 2010b].

Assuming that the scores are log-likelihood ratios, we will essentially plot act-DCF and min-DCF as a function of operating point. For every operating point, the value of act-DCF is then computed using analytically set threshold, see (**??**).

It is ensured, that min-DCF $\leq$ act-DCF. If the two functions are close, the scores are good log-likelihood ratios and the calibration is good, if they are very different, the calibration is bad. A good application-independent detector can be recognized by having a good calibration for all possible applications. Examples of these plots for a case of good and bad calibration are given in Figure 3.2.

3.5 Calibrating the Scores

So far, we were speaking about the calibration only in the sense that the scores provided by the system are good log-likelihood ratios. In practice, it is often not the case for many systems and the outputs of such systems have to be transformed into log-likelihood ratios. This is usually achieved by means of some transformation function $f(s)$, which is monothonic increasing. Usually a linear function

$$f(s) = a\,s + b, \tag{3.16}$$

is enough to convert scores into log-likelihod ratios. Parameters a and b are typically found by optimizing a cross-entropy objective function on a supervised set of development scores over a wide range of operating points [Brümmer, 2010b]. We will further discuss the cross-entropy objective function in Section 9.3.1.

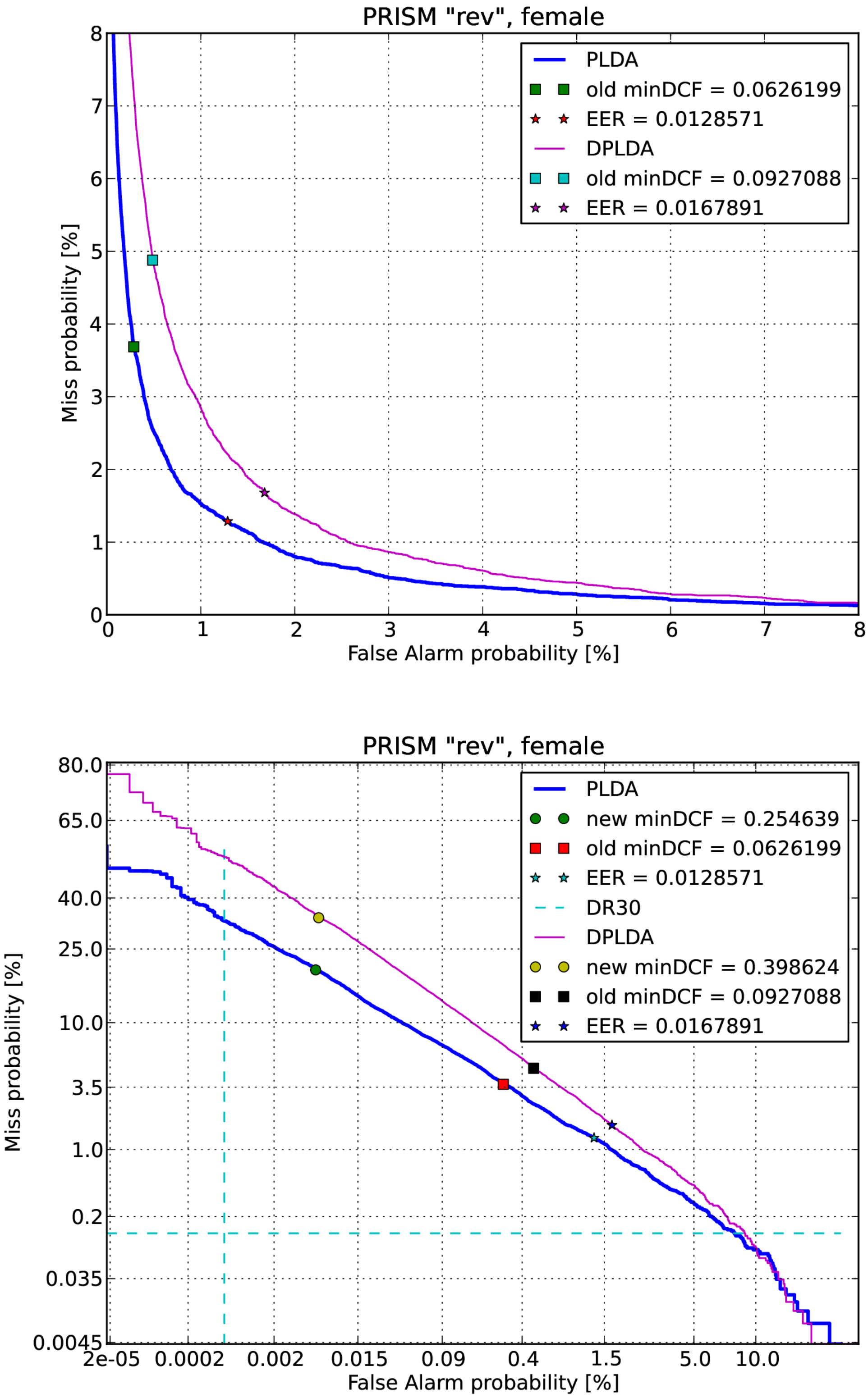


Figure 3.1: ROC (top) and DET (bottom) curves comparing two different techniques (PLDA and DPLDA). The three markers in each DET curve correspond to the new min-DCF (circle), the old min-DCF (square), and the EER (star). The region, where the values are statistically significant according to the Doddington's rule of 30 (DR30) is marked by the dashed blue lines.

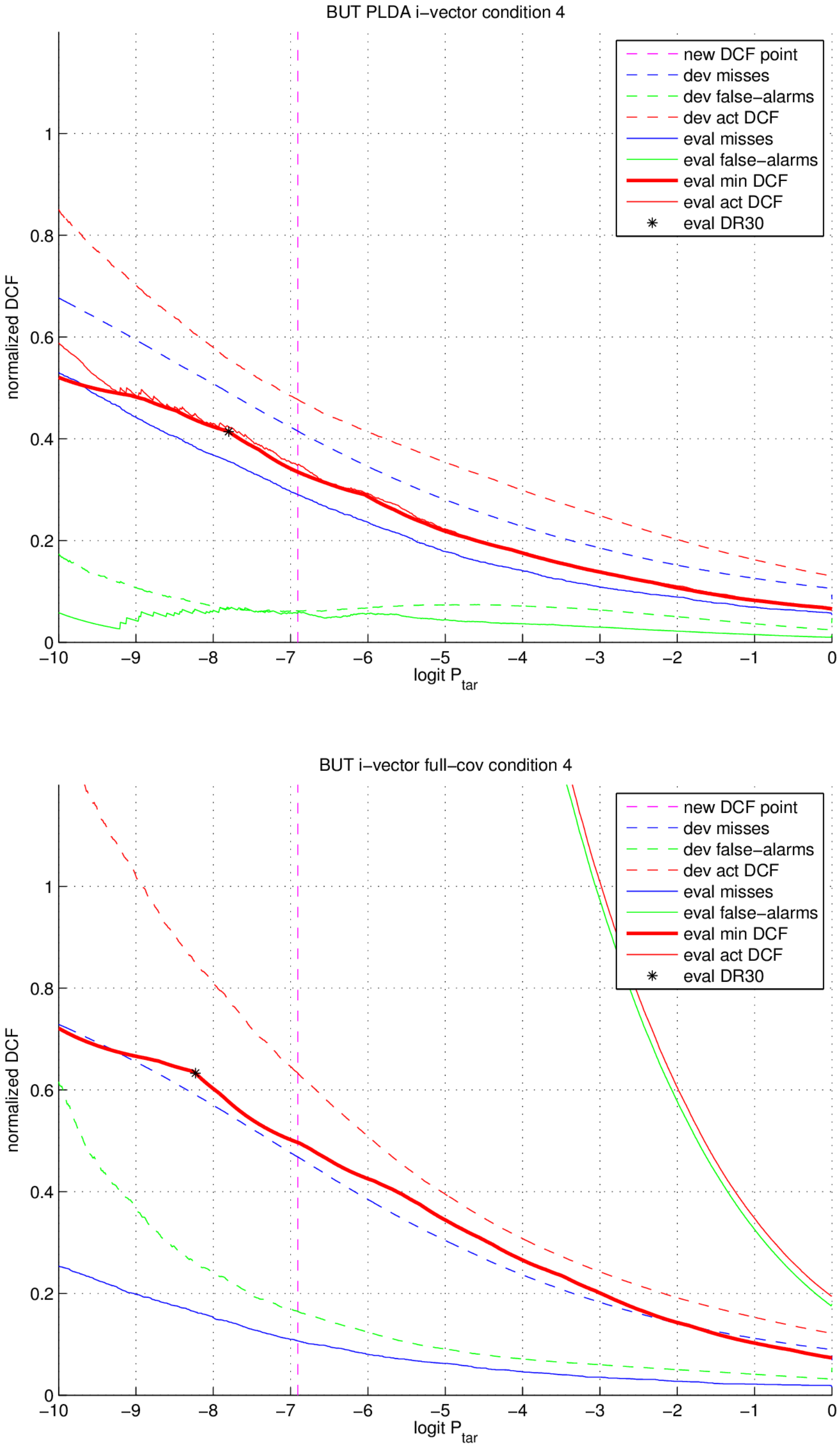


Figure 3.2: Normalized Bayes error-rate plots for two systems submitted to the NIST SRE 2010 evaluations. The top system based on PLDA and i–vectors represents an excellent calibration. The bottom system based on the same i–vectors, but with cosine distance scoring has a very bad calibration. *Eval* denotes the evaluation database and *dev* the development database. Miss and normalized false alarm rates are also shown separately. DR30 corresponds to the Doddington's rule of 30. To the left of this point, there are fewer than 30 false alarms. The vertical dashed magenta line represents the operating point of $\mathrm{DCF}_{\mathrm{new}}$ with $p(H_s) = 0.001$.

Chapter 4

Gaussian Mixture Modeling of Acoustic Features

Gaussian Mixture Models (GMM) are a family of mixture models where the probability density function (PDF) for each mixture (component) is a Gaussian distribution. As the GMMs are commonly used to model the probability distribution of features in bio-metric systems, naturally, they are widely used also in all fields of speech processing. This includes speaker recognition [Reynolds et al., 2000], as well as language identification (e.g. [Torres-Carrasquillo et al., 2002]), LVCSR (e.g. [Young et al., 2006]) and others. Generally, GMM parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm or, given an already pre-trained GMM model, by Maximum *A Posteriori* (MAP) re-estimation.

The main role of the GMM is to estimate an underlying distribution of acoustic features extracted from speech segments and inherently model the hidden classes, which are formed by individual speakers, various acoustic channels or some other common properties. This ability of unsupervised modeling of classes is later exploited by a supervised algorithm focused on extracting the information about the distributions of particular classes, e.g. those associated with speaker identities.

Let us define a speech segment as a set of F-dimensional acoustic features: $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\tau\}$. A GMM [Bishop, 2006] is then defined as a weighted sum (mixture) of a set of C multivariate normal distributions of the form:

$$p(\mathbf{x}|\mathcal{G}) = \sum_{c=1}^{C} w^{(c)} \mathcal{N}\left(\mathbf{x}; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)}\right), \tag{4.1}$$

where $p(\mathbf{x}|\mathcal{G})$ is the probability of $\mathbf{x}$ given the GMM model $\mathcal{G}$ with C mixture components and $w^{(c)}$ are individual mixture weights, also called mixing coefficients, satisfying the constraints that $w^{(c)} \geq 0$ and $\sum_{c=1}^{C} w^{(c)} = 1$. $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)})$ is an F-variate Gaussian component PDF with mean $\boldsymbol{\mu}^{(c)}$ and covariance matrix $\boldsymbol{\Sigma}^{(c)}$:

$$\mathcal{N}\left(\mathbf{x}; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)}\right) = \frac{1}{(2\pi)^{F/2} |\boldsymbol{\Sigma}^{(c)}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}^{(c)})^{\mathrm{T}} \boldsymbol{\Sigma}^{(c)^{-1}} (\mathbf{x}-\boldsymbol{\mu}^{(c)})}. \tag{4.2}$$

The whole GMM $\mathcal{G}$ is then represented by parameters

$$\lambda = \left\langle w^{(c)}, \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)} \right\rangle \qquad \text{with} \qquad c = 1 \ldots C, \tag{4.3}$$

or more conveniently by the supervectors and the matrix of stacked parameters as:

$$\lambda = \langle \mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma} \rangle = \left\langle \begin{bmatrix} w^{(1)} \\ \vdots \\ w^{(C)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\mu}^{(1)} \\ \vdots \\ \boldsymbol{\mu}^{(C)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}^{(1)} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{\Sigma}^{(C)} \end{bmatrix} \right\rangle . \tag{4.4}$$

It should be noted, that the covariance matrices can be full rank or constrained to be diagonal. Sometimes, the parameters can be shared among the Gaussian components. In general, the configuration with full covariance matrices needs more training data to properly estimate all the parameters. Often, a GMM with larger amount of components with diagonal covariance matrices is used instead of the configuration with full rank covariance matrices.

For evaluating the GMM model given the data, and therefore also for estimating its parameters, it is necessary to define the quantities associated with individual GMM components. Having observed the data point $\mathbf{x}_i$, posterior probabilities $p(c|\mathbf{x}_i)$, also referred to as *occupation probabilities* and shortly denoted as $\gamma_i^{(c)}$, can be computed using the Bayes rule:

$$\gamma_i^{(c)} = \frac{w^{(c)}\mathcal{N}\left(\mathbf{x}_i; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)}\right)}{\sum_{c=1}^{C} w^{(c)}\mathcal{N}\left(\mathbf{x}_i; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)}\right)} . \tag{4.5}$$

The configuration of the posterior probabilities for each feature vector is referred to as the *alignment* of the data to the mixture components. In this text, we will always assume, that the alignment of the feature vectors to Gaussian components is always based on Universal Background Model (UBM).

It is also convenient to define Baum-Welch statistics. Having our speech segment $\mathcal{X}$ which consists of $i = 1 \ldots \tau$ feature vectors of dimensionality F and the alignment of each feature vector $\mathbf{x}_i$ defined by (4.5), the Baum-Welsch [Kenny et al., 2007] statistics are defined as

$$N^{(c)} = \sum_{i=1}^{\tau} \gamma_i^{(c)} \tag{4.6}$$

$$\mathbf{f}^{(c)} = \sum_{i=1}^{\tau} \gamma_i^{(c)} \mathbf{x}_i \tag{4.7}$$

$$\mathbf{S}^{(c)} = \sum_{i=1}^{\tau} \gamma_i^{(c)} \mathbf{x}_i {\mathbf{x}_i}^{\mathrm{T}} . \tag{4.8}$$

We refer to these as the zero-, the first-, and the second-order statistics (or cumulants), respectively. For the simplification of the derivations, the statistics centered around the UBM mean are defined as

$$\tilde{\mathbf{f}}^{(c)} = \mathbf{f}^{(c)} - N^{(c)} \boldsymbol{\mu}^{(c)} \tag{4.9}$$

$$\tilde{\mathbf{S}}^{(c)} = \mathbf{S}^{(c)} - \mathbf{f}^{(c)}\boldsymbol{\mu}^{(c)\mathrm{T}} - \boldsymbol{\mu}^{(c)}\mathbf{f}^{(c)\mathrm{T}} + N^{(c)}\boldsymbol{\mu}^{(c)}\boldsymbol{\mu}^{(c)\mathrm{T}}. \tag{4.10}$$

For further simplification, the statistics can be stacked into the form of supervector and matrices as:

$$\mathbf{N} = \begin{bmatrix} N^{(1)}\mathbf{I} & 0 & \cdots & 0 \\ 0 & N^{(2)}\mathbf{I} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & N^{(C)}\mathbf{I} \end{bmatrix}$$

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}^{(1)} \\ \vdots \\ \mathbf{f}^{(C)} \end{bmatrix} \tag{4.11}$$

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}^{(1)} & 0 & \cdots & 0 \\ 0 & \mathbf{S}^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{S}^{(C)} \end{bmatrix},$$

where the identity matrices in (4.11) have the same dimensionality as the feature vector. Stacked centered statistics $\tilde{\mathbf{f}}$ and $\tilde{\mathbf{S}}$ are created according to the same scheme as their non-centered versions.

4.1 Maximum Likelihood Estimate of Parameters

Given enough training data and some initial GMM configuration $\lambda^{(0)}$, we want to estimate the new parameters, which best match the underlying distribution of the data. A possible approach is to perform a Maximum-Likelihood (ML) estimate [Reynolds and Rose, 1995, Bishop, 2006] and search for the solution of

$$\lambda_{\mathrm{ML}} = \arg\max_{\lambda} p(\mathcal{X}|\lambda). \tag{4.12}$$

Assuming the statistical independence of the frames/feature vectors, the likelihood of the data $\mathcal{X}$, given the model parameters λ, is given as

$$p(\mathcal{X}|\lambda) = \prod_{i=1}^{\tau} \mathcal{G}(\mathbf{x}_i; \lambda). \tag{4.13}$$

Usually, the logarithm of the likelihood is required for evaluating the model and estimating the parameters. Its basic form is given as

$$\log p(\mathcal{X}|\lambda) = \sum_{i=1}^{\tau} \log \sum_{c=1}^{C} w^{(c)} \mathcal{N}\left(\mathbf{x}_i; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)}\right). \tag{4.14}$$

For any choice of distributions $q_i(c)$ over the Gaussian components, we can rewrite this likelihood as

$$\begin{aligned}\log p(\mathcal{X}|\lambda) = \sum_{i=1}^{\tau} \log p(\mathbf{x}_i|\lambda) &= \sum_{i=1}^{\tau} \underbrace{\sum_{c=1}^{C} q_i(c)}_{1} \log \frac{p(\mathbf{x}_i, c|\lambda)}{p(c|\mathbf{x}_i, \lambda)} \frac{q_i(c)}{q_i(c)} \\ &= \sum_{i=1}^{\tau} \left[\sum_{c=1}^{C} q_i(c) \log \left(w^{(c)} \mathcal{N} \left(\mathbf{x}_i; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)} \right) \right) \right. \\ &\left. - \sum_{c=1}^{C} q_i(c) \log q_i(c) + \sum_{c=1}^{C} q_i(c) \log \frac{q_i(c)}{\gamma_i^{(c)}} \right], \end{aligned} \tag{4.15}$$

where the last term

$$\sum_{c=1}^{C} q_i(c) \log \frac{q_i(c)}{\gamma_i^{(c)}} = \mathrm{D_{KL}}(q_i(c) \| \gamma_i^{(c)}) \tag{4.16}$$

corresponds to the Kullback-Leibler (KL) divergence between $q_i(c)$ and the posterior distribution $p(c|\mathbf{x}_i, \lambda) = \gamma_i^{(c)}$. Hence, if we set $q_i(c)$ to the true posterior $\gamma_i^{(c)}$, the KL divergence vanishes and the likelihood can be expressed as

$$\log p(\mathcal{X}|\lambda) = \sum_{i=1}^{\tau} \left[\sum_{c=1}^{C} \gamma_i^{(c)} \log \left(w^{(c)} \mathcal{N} \left(\mathbf{x}_i; \boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)} \right) \right) - \sum_{c=1}^{C} \gamma_i^{(c)} \log \gamma_i^{(c)} \right]. \tag{4.17}$$

Using the Baum-Welch statistics, we can further rewrite the log-likelihood [Kenny et al., 2004] and get

$$\begin{aligned}\log p(\mathcal{X}|\lambda) = \sum_{c=1}^{C} &\left[N^{(c)} \log \frac{1}{(2\pi)^{F/2} |\boldsymbol{\Sigma}^{(c)}|^{1/2}} \right. \\ &\left. - \frac{1}{2} \mathrm{tr} \left(\boldsymbol{\Sigma}^{(c)^{-1}} \left(\mathbf{S}^{(c)} - \mathbf{f}^{(c)} \boldsymbol{\mu}^{(c)^{\mathrm{T}}} - \boldsymbol{\mu}^{(c)} \mathbf{f}^{(c)^{\mathrm{T}}} + N^{(c)} \boldsymbol{\mu}^{(c)} \boldsymbol{\mu}^{(c)^{\mathrm{T}}} \right) \right) \right] \\ &- \sum_{i=1}^{\tau} \sum_{c=1}^{C} \gamma_i^{(c)} \log \frac{\gamma_i^{(c)}}{w^{(c)}}, \end{aligned} \tag{4.18}$$

which is the correct likelihood, if the statistics were collected with the true posterior distribution $\gamma_i^{(c)}$. If the true posterior distribution is not available and is provided via different model, e.g. UBM, then this function serves as an approximation and a lower-bound of the correct likelihood, since the omitted KL divergence is always non-negative.

Unfortunately, direct optimization of the parameters given the data is analytically intractable. However, ML estimates of the parameters can be obtained iteratively by the means of EM algorithm [Dempster et al., 1977, Bishop, 2006].

For the E-step of the EM algorithm, the auxiliary function can be constructed from

(4.18) as

$$\begin{aligned}\mathcal{Q}_{\text{GMM}}(\lambda,\lambda^{(0)}) = &\sum_{c=1}^{C}\Bigg[N_{\lambda_0}^{(c)}\log\frac{1}{(2\pi)^{F/2}|\mathbf{\Sigma}^{(c)}|^{1/2}}\\ &-\frac{1}{2}\text{tr}\left(\mathbf{\Sigma}^{(c)-1}\left(\mathbf{S}_{\lambda_0}^{(c)}-\mathbf{f}_{\lambda_0}^{(c)}\boldsymbol{\mu}^{(c)\mathrm{T}}-\boldsymbol{\mu}^{(c)}\mathbf{f}_{\lambda_0}^{(c)\mathrm{T}}+N_{\lambda_0}^{(c)}\boldsymbol{\mu}^{(c)}\boldsymbol{\mu}^{(c)\mathrm{T}}\right)\right)\Bigg]\\ &+\sum_{c=1}^{C}\log w^{(c)}.\end{aligned} \tag{4.19}$$

By fixing the alignment of the data using the current model estimate $\lambda^{(0)}$, we obtain $\gamma_{i\lambda_0}^{(c)}$ and collect the statistics $\{N_{\lambda_0}^{(c)}, \mathbf{f}_{\lambda_0}^{(c)}, \mathbf{S}_{\lambda_0}^{(c)}\}$. In the M-step of the algorithm, the new ML estimate of parameters is then computed as

$$\lambda_{\text{ML}} = \arg\max_{\lambda}\mathcal{Q}_{\text{GMM}}(\lambda,\lambda^{(0)}), \tag{4.20}$$

for which the update formulas are given as:

$$\begin{aligned}\boldsymbol{\mu}_{\text{ML}}^{(c)} &= \frac{1}{N^{(c)}}\mathbf{f}^{(c)}\\ \mathbf{\Sigma}_{\text{ML}}^{(c)} &= \frac{1}{N^{(c)}}\mathbf{S}^{(c)}-\boldsymbol{\mu}_{\text{ML}}^{(c)}\boldsymbol{\mu}_{\text{ML}}^{(c)\ \mathrm{T}}\\ w_{\text{ML}}^{(c)} &= \frac{N^{(c)}}{\tau}.\end{aligned} \tag{4.21}$$

Repeating the E and M steps guarantees not to decrease the likelihood and iterating is usually stopped when the likelihood increase in two consecutive iterations is smaller than some convergence threshold. For more detailed derivations following roughly our notation, we refer the kind reader to [Glembek, 2012].

ML can be safely used only when sufficient amount of data is available (see e.g. [Burget et al., 2007]) and therefore this approach is used for training the UBM. For estimating speaker models when considering the asymmetrical SRE approach (see Section 1.2), a small amount of data is given for training. In this case a common practice is to re-estimate only the UBM means by performing MAP adaptation [Reynolds et al., 2000]. Therefore, the speaker model, is given strictly by $\boldsymbol{\mu}$, while $\boldsymbol{w}$ and $\mathbf{\Sigma}$ are shared among all models and are taken from the UBM.

4.2 MAP Adaptation

Another approach to estimate the GMM parameters is to use the *maximum a-posteriori* criterion (MAP). This approach is often used when we already have robustly estimated *a-priori* information (in our case $p(\lambda)$) about the process whose parameters we want to estimate. The a-priori information can come from the scientific assumptions or, as in our case, it can be estimated from previously observed data. The source of an a-priori knowledge in speech processing is usually the ML-estimated UBM.

Generally, the parameters are computed as

$$\lambda_{\mathrm{MAP}} = \arg\max_{\lambda} p(\lambda|\mathcal{X}), \tag{4.22}$$

where $p(\lambda|\mathcal{X})$ is the posterior probability for the parameters λ given the input data $\mathcal{X}$:

$$p(\lambda|\mathcal{X}) = \frac{p(\mathcal{X}|\lambda)p(\lambda)}{p(\mathcal{X})} \propto p(\mathcal{X}|\lambda)p(\lambda). \tag{4.23}$$

Inserting (4.23) into (4.22), the MAP estimation is then given as

$$\lambda_{\mathrm{MAP}} = \arg\max_{\lambda} p(\mathcal{X}|\lambda)p(\lambda). \tag{4.24}$$

As already mentioned, this approach is helpful if very little data is available and more importantly if we have a good prior. Comparing (4.24) and (4.12), we can see ML as a special case of MAP, where flat priors are considered. Having no information about the prior can yield a good parameter estimate only for large training data sets.

Let us now demonstrate, how the UBM can be used as a source of prior for "adapting" the $\boldsymbol{\mu}$. Again, we only work with $\boldsymbol{\mu}$, as it is a common practice in SRE, letting the rest of the parameters being shared with the UBM. For the MAP estimate of weights and covariance matrices, see [Gauvain and Lee, 1994, Reynolds et al., 2000].

Relevance MAP mean adaptation as proposed in [Reynolds et al., 2000] can be computed as

$$\boldsymbol{\mu}^{(c)}_{\mathrm{MAP}} = \beta^{(c)}\boldsymbol{\mu}^{(c)}_{\mathrm{ML}} + \left(1 - \beta^{(c)}\right)\boldsymbol{\mu}^{(c)}_{\mathrm{UBM}}, \tag{4.25}$$

with

$$\beta^{(c)} = \frac{N^{(c)}}{N^{(c)} + r}, \tag{4.26}$$

where $\boldsymbol{\mu}^{(c)}_{\mathrm{ML}}$ (see (4.37)) is the UBM mean, ML re-estimated in a single iteration of the EM algorithm, $\boldsymbol{\mu}^{(c)}_{\mathrm{UBM}}$ is the original UBM mean and $N^{(c)}$ are zero-order statistics. The adaptation constant r, often referred to as a *relevance factor*, acts as a trade-off between the ML estimate and the UBM. The value of this constant is chosen by the user. It will be shown in Section 4.3 that this update corresponds to the MAP estimate with a specific choice of prior imposed on GMM means.

4.2.1 Speaker Verification Using Relevance MAP

If we want to do speaker verification considering the MAP-estimated mean $\boldsymbol{\mu}_s$ from the enrollment data as a model for a speaker s, we obtain the score as the Log-Likelihood Ratio (LLR) between the speaker model and the UBM log-likelihood for the test utterance $\mathcal{X}_{\mathrm{test}}$. By evaluating likelihood (4.14) for both GMMs, we get:

$$LLR = \log p(\mathcal{X}|\lambda_s) - \log p(\mathcal{X}|\lambda), \tag{4.27}$$

where, with $c = 1 \dots C$, $\lambda = \left\langle \boldsymbol{\mu}^{(c)}_{\mathrm{UBM}}, \boldsymbol{\Sigma}^{(c)}_{\mathrm{UBM}}, w^{(c)}_{\mathrm{UBM}} \right\rangle$ are the parameters of the UBM and $\lambda_s = \left\langle \boldsymbol{\mu}^{(c)}_{s}, \boldsymbol{\Sigma}^{(c)}_{\mathrm{UBM}}, w^{(c)}_{\mathrm{UBM}} \right\rangle$ are the parameters representing the speaker model.

4.3 Latent Variable Models for Speaker Recognition

In this Section, we will describe essential techniques based on Factor Analysis [Bishop, 2006]. These techniques build upon the MAP estimate of the speaker-dependent GMM, while taking into account either inter- or intra-session variability or both of them at the same time. To study the problematic in detail, we refer the reader to the following publications [Kenny, 2005, Kenny et al., 2007, Kenny et al., 2005a].

Let us begin with a brief description of MAP adaptation in terms of hidden variable models by following [Kenny, 2005]. Continuing with the notation of GMM from the previous section, we will define the speaker-dependent supervector $\mathbf{g}(s)$ as a latent variable model for speaker s as

$$\mathbf{g}(s) = \boldsymbol{\mu} + \mathbf{D}\mathbf{z}(s). \tag{4.28}$$

The speaker-dependent supervector is distributed according to $\mathbf{g} \sim (\boldsymbol{\mu}, \mathbf{D}\mathbf{D}^{\mathrm{T}})$ and a $CF \times S$ matrix $\mathbf{D}$ acts as a prior on the UBM mean supervector $\boldsymbol{\mu}$. Latent variable $\mathbf{z}(s)$ is a S-dimensional speaker-dependent hidden vector distributed according to the standard normal distribution, $\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$. The S in the dimensionalities of the variabilities denotes an arbitrary positive number and will be discussed later in the end of Section 4.3.1.

The log-likelihood of data and the hidden variable is based on the general GMM log-likelihood function as defined in Section 4.1. We will assume fixed data alignment [Kenny, 2005] and represent the log-likelihood by the means of the Baum-Welch statistics collected using UBM. As already discussed in the previous section, this is an approximated log-likelihood acting as a lower-bound to the real log likelihood. Using the Universal Background Model to collect the statistics for all observations $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\tau\}$ corresponding to the speaker s, we get

$$\begin{aligned} \log p(\mathcal{X}|\mathbf{D}, \mathbf{z}) &= G + H(\mathbf{z}), \\ G &= \sum_{c=1}^{C} \left(N_{\mathcal{X}}^{(c)} \log \frac{1}{(2\pi)^{F/2} |\boldsymbol{\Sigma}^{(c)}|^{1/2}} \right) - \frac{1}{2} \mathrm{tr} \left(\boldsymbol{\Sigma}^{-1} \tilde{\mathbf{S}}_{\mathcal{X}} \right), \\ H(\mathbf{z}) &= \mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}_{\mathcal{X}} - \frac{1}{2} \mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \mathbf{N}_{\mathcal{X}} \mathbf{D} \mathbf{z}, \end{aligned} \tag{4.29}$$

where $\boldsymbol{\Sigma}$ is a block diagonal covariance matrix of the UBM composed as in (4.3), $\mathbf{N}_{\mathcal{X}}$, $\tilde{\mathbf{f}}_{\mathcal{X}}$ and $\tilde{\mathbf{S}}_{\mathcal{X}}$ are stacked zero-, first- and second-order centered statistics collected with the UBM according to (4.6), (4.9) and (4.10).

The joint log-likelihood of the observed data $\mathcal{X}$ and the hiden variable is given by

$$\begin{aligned} \log p(\mathcal{X}, \mathbf{z}|\mathbf{D}) &= \log p(\mathcal{X}|\mathbf{D}, \mathbf{z}) p(\mathbf{z}) \\ &= K_{\boldsymbol{\Sigma}} + (\mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}_{\mathcal{X}} - \frac{1}{2} \mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \mathbf{N}_{\mathcal{X}} \mathbf{D} \mathbf{z} - \frac{1}{2} \mathbf{z}^{\mathrm{T}} \mathbf{z}), \end{aligned} \tag{4.30}$$

where the term $K_{\boldsymbol{\Sigma}}$ is a constant (also referred to as a normalization term), which does not depend on $\mathbf{z}$ and $\mathbf{D}$. Leaving out the $K_{\boldsymbol{\Sigma}}$, the posterior of the hidden variable $\mathbf{z}$, given the data $\mathcal{X}$ observed for speaker s, is given as

$$\log p(\mathbf{z}|\mathcal{X}) \propto \log p(\mathcal{X}, \mathbf{z}) \propto \left(\mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}_{\mathcal{X}} - \frac{1}{2} \mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \mathbf{N}_{\mathcal{X}} \mathbf{D} \mathbf{z} - \frac{1}{2} \mathbf{z}^{\mathrm{T}} \mathbf{z} \right). \tag{4.31}$$

By completion of squares, the posterior for $\mathbf{z}$ is also Gaussian

$$p(\mathbf{z}|\mathcal{X}) \sim \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_{\mathbf{z}}, \boldsymbol{\Gamma}_{\mathbf{z}}^{-1}), \tag{4.32}$$

with precision matrix and mean given by

$$\boldsymbol{\Gamma}_{\mathbf{z}} = (\mathbf{D}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{N}_{\mathcal{X}}\mathbf{D} + \mathbf{I}) \tag{4.33}$$

$$\boldsymbol{\mu}_{\mathbf{z}} = \boldsymbol{\Gamma}_{\mathbf{z}}^{-1}\mathbf{D}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}_{\mathcal{X}}. \tag{4.34}$$

The mean of supervector posterior $p(\mathbf{g}|\mathcal{X})$ (i.e. its MAP estimate) is then given as

$$\begin{aligned}\hat{\mathbf{g}} &= \boldsymbol{\mu} + \mathbf{D}\boldsymbol{\mu}_{\mathbf{z}} \\ &= \boldsymbol{\mu} + \mathbf{D}(\mathbf{D}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{N}_{\mathcal{X}}\mathbf{D} + \mathbf{I})^{-1}\mathbf{D}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}_{\mathcal{X}} \\ &= \boldsymbol{\mu} + (\mathbf{N}_{\mathcal{X}} + \boldsymbol{\Sigma}(\mathbf{D}\mathbf{D}^{\mathrm{T}})^{-1})^{-1}\tilde{\mathbf{f}}_{\mathcal{X}}.\end{aligned} \tag{4.35}$$

By setting $\mathbf{D}\mathbf{D}^{\mathrm{T}} = \frac{\boldsymbol{\Sigma}}{r}$ (i.e. setting $\mathbf{D} = \mathrm{chol}\frac{\boldsymbol{\Sigma}}{r}$), we can rewrite the adapted model as

$$\hat{\mathbf{g}} = \boldsymbol{\mu} + (\mathbf{N}_{\mathcal{X}} + \mathbf{r})^{-1}\,\tilde{\mathbf{f}}_{\mathcal{X}}, \tag{4.36}$$

which corresponds to a heuristic prior $\mathbf{g} \sim \mathcal{N}(\boldsymbol{\mu}, \frac{\boldsymbol{\Sigma}}{r})$ imposed on the mean supervector, so we can write that

$$\begin{aligned}\hat{\mathbf{g}} &= \boldsymbol{\mu} + \frac{1}{\mathbf{N}_{\mathcal{X}} + r}\,\tilde{\mathbf{f}}_{\mathcal{X}} \\ &= \underbrace{\frac{r}{\mathbf{N}_{\mathcal{X}} + r}}_{1-\beta}\,\boldsymbol{\mu} + \underbrace{\frac{\mathbf{N}_{\mathcal{X}}}{\mathbf{N}_{\mathcal{X}} + r}}_{\beta}\,\underbrace{\boldsymbol{\mu}\,\tilde{\mathbf{f}}_{\mathcal{X}}\,\mathbf{N}_{\mathcal{X}}^{-1}}_{\boldsymbol{\mu}_{\mathrm{ML}}},\end{aligned} \tag{4.37}$$

which is in agreement with the relevance MAP adaptation formula (4.25).

4.3.1 Training Prior Hyper-Parameters

In the previous section, we discussed how to artificially supply a prior by means of another model (UBM). Now, we will describe how to train it from the data in a ML fashion. The training objective is to maximize the likelihood of the training data $p(\mathcal{X}|\mathbf{D}, \mathbf{z})$. Similarly to the GMM training, the ML estimate of the parameters can be obtained by means of EM algorithm [Brümmer, 2009]. While the other parameters $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{w}\}$ could be also re-estimated, here we will consider re-estimating only the matrix $\mathbf{D}$. Taking the $\mathbf{z}$ as a hidden variable, the EM auxiliary function is then constructed as

$$\mathcal{Q}(\mathbf{D}, \mathbf{D}_0) = \sum_s \langle \log p(\mathcal{X}_s, \mathbf{z}|\mathbf{D}_0)\rangle_{\mathbf{z}|\mathcal{X}_s, \mathbf{w}|\mathbf{D}_0}, \tag{4.38}$$

where $p(\mathcal{X}_s, \mathbf{z}|\mathbf{D}_0)$ is the joint probability of the observations $\mathcal{X}_s$ for speaker s. Considering that

$$p(\mathcal{X}_s, \mathbf{z}|\mathbf{D}) = \log p(\mathcal{X}_s|\mathbf{D}, \mathbf{z}) + \log p(\mathbf{z}) \tag{4.39}$$

and $p(\mathbf{z})$ being set to a standard normal distribution and kept fixed, there is no need to re-estimate parameters of $p(\mathbf{z})$, as any changes in the prior distribution can be equivalently accomplished by appropriately changing $\boldsymbol{\mu}$ and $\mathbf{D}$. Therefore, we can simplify the auxiliary function as

$$\mathcal{Q}(\mathbf{D}, \mathbf{D}_0) = \sum_s \langle \log p(\mathcal{X}_s | \mathbf{z}, \mathbf{D}_0) \rangle_{\mathbf{z}|\mathcal{X}_s, \mathbf{D}_0}. \tag{4.40}$$

By looking at the expression for the joint likelihood (4.30) and realizing that $K_{\boldsymbol{\Sigma}}$ does not depend on $\mathbf{D}$, we can further express the auxiliary function as

$$\begin{aligned} \mathcal{Q}(\mathbf{D}, \mathbf{D}_0) &= \sum_s \left\langle \mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}_{\mathcal{X}_s} - \frac{1}{2} \mathbf{z}^{\mathrm{T}} \mathbf{D}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \mathbf{N}_{\mathcal{X}_s} \mathbf{D} \mathbf{z} \right\rangle_{\mathbf{z}|\mathcal{X}_s, \mathbf{D}_0} \\ &= \sum_s \operatorname{tr} \left[\boldsymbol{\Sigma}^{-1} \left(\tilde{\mathbf{f}}_{\mathcal{X}_s} \langle \mathbf{z} \rangle \mathbf{D}^{\mathrm{T}} - \frac{1}{2} \mathbf{N}_{\mathcal{X}_s} \mathbf{D} \langle \mathbf{z} \mathbf{z}^{\mathrm{T}} \rangle \mathbf{D}^{\mathrm{T}} \right) \right], \end{aligned} \tag{4.41}$$

where the expectations are taken over $\mathbf{z}|\mathcal{X}_s, \mathbf{D}_0$. Now, in order to minimize the auxiliary function, we can take its derivative with respect to $\mathbf{D}$ and set it to zero:

$$\frac{\partial}{\partial \mathbf{D}} \sum_s \operatorname{tr} \left[\boldsymbol{\Sigma}^{-1} \left(\tilde{\mathbf{f}}_{\mathcal{X}_s} \langle \mathbf{z} \rangle \mathbf{D}^{\mathrm{T}} - \frac{1}{2} \mathbf{N}_{\mathcal{X}_s} \mathbf{D} \langle \mathbf{z} \mathbf{z}^{\mathrm{T}} \rangle \mathbf{D}^{\mathrm{T}} \right) \right] = \mathbf{0}, \tag{4.42}$$

which gives

$$\sum_s \boldsymbol{\Sigma}^{-1} \left(\tilde{\mathbf{f}}_{\mathcal{X}_s} \langle \mathbf{z} \rangle - \mathbf{N}_{\mathcal{X}_s} \mathbf{D} \langle \mathbf{z} \mathbf{z}^{\mathrm{T}} \rangle \right) = \mathbf{0}. \tag{4.43}$$

We need to solve the linear system

$$\mathbf{D}^{(c)} \sum_s N_{\mathcal{X}_s}^{(c)} \langle \mathbf{z}_{\mathcal{X}_s} \mathbf{z}_{\mathcal{X}_s}^{\mathrm{T}} \rangle = \sum_s \tilde{\mathbf{f}}_{\mathcal{X}_s}^{(c)} \langle \mathbf{z}_{\mathcal{X}_s}^{\mathrm{T}} \rangle, \tag{4.44}$$

where c is spanning the rows of the matrices corresponding to individual UBM components. The expectation over the hidden variable $\langle \mathbf{z} \rangle$ is given as a mean of the posterior distribution of $\mathbf{z}$ given the $\mathbf{D}_0$ (see (4.34)) and $\langle \mathbf{z}\mathbf{z}^{\mathrm{T}} \rangle = \langle \mathbf{z} \rangle \langle \mathbf{z}^{\mathrm{T}} \rangle + \boldsymbol{\Gamma}_{\mathbf{z}}^{-1}$, where $\boldsymbol{\Gamma}_{\mathbf{z}}^{-1}$ is the covariance matrix (see (4.33)) of the posterior of $\mathbf{z}$ given $\mathbf{D}_0$. Finally, the closed-form solution for computing the hyper-parameters is :

$$\mathbf{D}^{(c)} = \sum_s \left[\tilde{\mathbf{f}}_{\mathcal{X}_s}^{(c)} \boldsymbol{\mu}_{\mathbf{z}\mathcal{X}_s} (N_{\mathcal{X}_s}^{(c)} (\boldsymbol{\mu}_{\mathbf{z}\mathcal{X}_s} \boldsymbol{\mu}_{\mathbf{z}\mathcal{X}_s}^{\mathrm{T}} + \boldsymbol{\Gamma}_{\mathbf{z}\mathcal{X}_s}^{-1}))^{-1} \right]. \tag{4.45}$$

The framework described in this section allows for setting different dimensionalities and constraints for $\mathbf{D}$. In theory, we could take $\mathbf{D}$ as a full $CF \times CF$ matrix. This would be impractical, since the amount of parameters to train would be very large. For this reason, $\mathbf{D}$ is often constrained to be diagonal or low rank. Taking $\mathbf{D}$ as a low-rank $CF \times S$ matrix constraints the speaker-dependent supervector to lie in a S-dimensional subspace, which is a widely used approach. The use of the subspace modeling will be shown in the following sections.

4.3.2 Eigenvoice Adaptation

The main idea behind the eigenvoice adaptation [Nguyen et al., 2000, Kenny et al., 2005a, Kenny et al., 2003] is to constrain the speaker-dependent supervectors to lie in a low-dimensional subspace spanned by M bases. This technique is effective if low amount of enrollment data is available for individual speakers, as with $M \ll CF$ the amount of parameters of the model is greatly reduced. Mathematically, this model can be written as

$$\mathbf{g}(s) = \boldsymbol{\mu} + \mathbf{U}\mathbf{y}(s), \tag{4.46}$$

where $\mathbf{U}$ is a low rank $CF \times M$ matrix. The columns of this matrix are called eigenvoices, because historically in ASR [Nguyen et al., 2000] they corresponded to M largest eigenvalues of the covariance matrix of the supervectors for the speaker population. This model corresponds to the classical MAP from the previous section and $\mathbf{y}$ is then speaker-dependent latent variable with standard normal prior distribution. However, if we compare $\mathbf{y}(s)$ and a latent variable from the classical MAP $\mathbf{z}(s)$, we see that the dimensionality of $\mathbf{y}(s)$ is much smaller. To obtain the posterior distribution of the latent variable, we can follow the same steps as in the previous section and get

$$p(\mathbf{y}|\mathcal{X}) \sim \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}_{\mathbf{y}}, \boldsymbol{\Gamma}_{\mathbf{y}}^{-1}), \tag{4.47}$$

with precision matrix and mean given by

$$\boldsymbol{\Gamma}_{\mathbf{y}} = (\mathbf{U}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{N}_{\mathcal{X}}\mathbf{U} + \mathbf{I}) \tag{4.48}$$

$$\boldsymbol{\mu}_{\mathbf{y}} = \boldsymbol{\Gamma}_{\mathbf{y}}^{-1}\mathbf{U}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}_{\mathcal{X}}. \tag{4.49}$$

Also the derivations for re-estimation of hyper-parameters can be obtained by following the same steps as in the previous section.

4.3.3 Channel Adaptation

So far, in previously described techniques, we did not consider the effects caused by a channel mismatch of individual recordings. Many different techniques for dealing with this unwanted *inter-session* (or also often called *channel*) variability have been introduced in the field of speaker recognition and in speech processing in general. Compensating for the effect caused by observing the data for a particular speaker under various acoustic conditions can yield substantial improvements. For speaker recognition, this compensation is especially effective, if we observe most of the possible channels during the training of the adaptation model, e.g. observing the data from microphones that will be used to record the target data. Then, at the test phase, when the enrollment and test data come from different microphones, the channel mismatch will be eliminated, which will improve performance of the system. The effect of channel mismatch can be seen Figure 4.1, where a particular system was evaluated on the NIST SRE 2008 interview condition, where multiple microphones were used to record the data.[1] It should be noted that the system presented in the Figure 4.1 is already trained with eigenchannel adaptation (see later in this section). We can see that the application of eigenchannel adaptation did not completely solve the channel mismatch problem, but it has narrowed the gap between the two scenarios (matched and mismatched channel in enroll and test). The channel compensation can be performed at different steps. At the feature level [Openshaw and Masan, 1994], usually cepstral mean and variance normalization (CMN, CVN) is performed as the cepstral features are used most often in speech processing.

[1] The figure is a proprietary image of Lukáš Burget and was used with his kind permission.

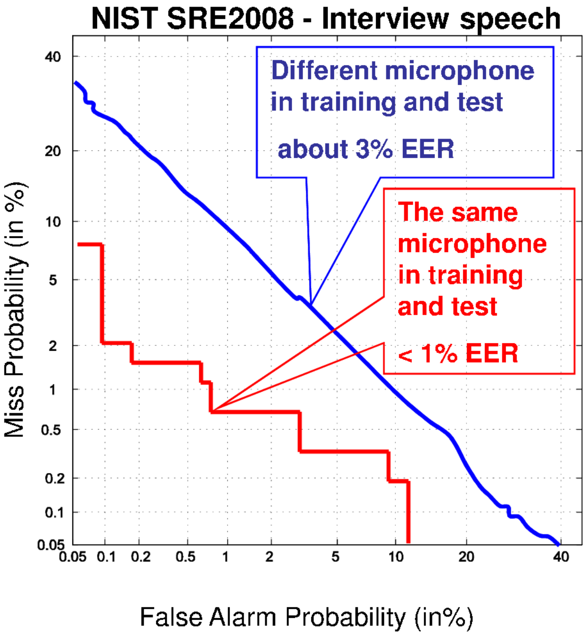


Figure 4.1: Example of a channel mismatch effect. The red DET curve shows a test case, when the same microphone was used for speaker model training and for test. The blue curve, however, shows the deterioration in performance when two different microphones are used.

Another approach based on a channel detector and a subsequent selection of channel-dependent model for feature adaptations was introduced in [Reynolds, 2003]. This technique is known as *feature mapping* and is based on transforming feature vectors into a channel-independent space. The transformation is trained on a set of channel-dependent models. A downside of this approach resides in the reliance on a channel detector — a closed-set multi-class classifier. Obviously, if the test utterance comes from an unseen channel (or unseen combination of known channel effects), the detector is likely to select a wrong model making the whole adaptation ineffective or even detrimental.

Apart from techniques operating directly in the feature space a model-level compensation using already introduced Factor Analysis was proposed in [Kenny and Dumouchel, 2004] and subsequently used in the NIST SRE in [Brümmer, 2004, Vogt et al., 2005]. In contrast to the eigenvoice adaptation, this technique is based on finding a subspace corresponding to the unwanted variability. Formally, we can write the model as

$$\mathbf{g}(s) = \boldsymbol{\mu}(s) + \mathbf{c}_{\mathcal{X}}(s), \tag{4.50}$$

where $\boldsymbol{\mu}(s)$ is the channel-independent speaker model and $\mathbf{c}_{\mathcal{X}}(s)$ corresponds to the supervector defining the channel offset given the utterance $\mathcal{X}$ for speaker s. The channel offset (or the channel component) is constrained to lie in a subspace defined as

$$\mathbf{c}_{\mathcal{X}}(s) = \mathbf{V}\mathbf{x}_{\mathcal{X}}(s), \tag{4.51}$$

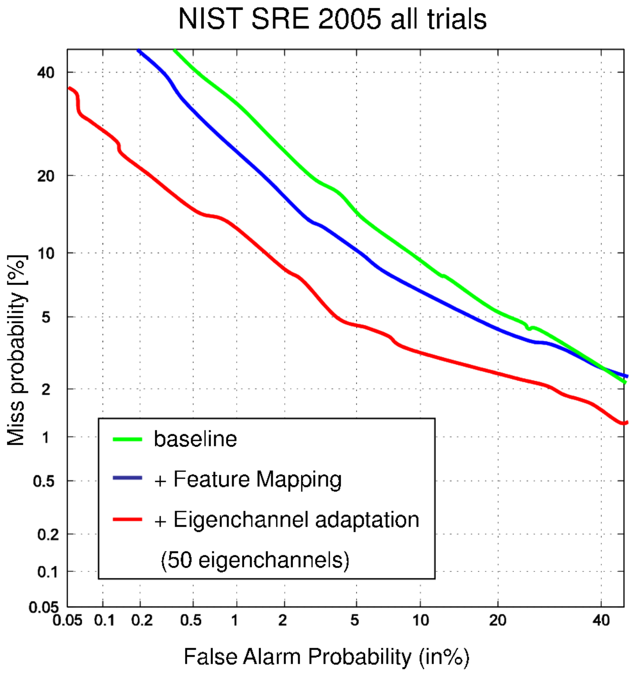


Figure 4.2: Comparison of the different channel compensation techniques on the complete NIST SRE 2005 data. We see the superior performance of the eigenchannel adaptation with respect to the Relevance-MAP baseline.

where $\mathbf{V}$ is a $CF \times R$ matrix with $R \ll CF$ and $\mathbf{x}_{\mathcal{X}}(s)$ — vector of *channel factors*, is a standard normal distributed hidden variable. In practice, the test data for individual speakers are used to adapt their models in the eigenchannel subspace. Solution to the posterior and hyper-parameter updates can be again obtained by following the steps in Section 4.3.

The effect of eigenchannel adaptation is shown in the Figure 4.2, on an exemplary system of BUT (Brno University of Technology) in the NIST SRE 2005.[2] For a thorough analysis on the NIST data, see e.g. [Burget et al., 2007].

4.3.4 Joint Factor Analysis

Joint Factor Analysis (JFA) is a GMM supspace modeling techique considering both speaker and channel variabilities [Kenny et al., 2007]. We can see it as a combination of eigenvoice and eigenchannel MAP into a single model, where the speaker- and channel-dependent supervector $\mathbf{g}_{\mathcal{X}}(s)$ can be factorized as

$$\mathbf{g}_{\mathcal{X}}(s) = \boldsymbol{\mu} + \mathbf{U}\mathbf{y}(s) + \mathbf{V}\mathbf{x}_{\mathcal{X}}(s) + \mathbf{D}\mathbf{z}(s), \tag{4.52}$$

where $\boldsymbol{\mu}$ is a CF dimensional speaker- and channel-independent mean supervector, usually taken from the UBM, the $\mathbf{U}$ and $\mathbf{V}$ are low-rank matrices spanning the speaker and

[2] The figure is a proprietary image of Lukáš Burget and was used with his kind permission.

channel subspace, respectively. $\mathbf{D}$ is a diagonal full-rank matrix representing the residual variability. The low-dimensional hidden variable vectors $\mathbf{y}(s)$ are known as speaker factors, $\mathbf{x}_{\mathcal{X}}(s)$ as channel factors and $\mathbf{z}(s)$ as common factors. This model then allows for modeling all variabilities jointly.

However, the model has undergone a series of modifications and simplifications since it was first used (see e.g. [Kenny and Dumouchel, 2004, Kenny et al., 2005b]). As the joint training of all parameters is very computationally expensive (for theory see [Kenny, 2005]), various simplifications involve separate training of individual subspaces, while keeping the other fixed, see [Kenny et al., 2005b, Kenny et al., 2008, Burget et al., 2009]. For this kind of training, the framework previously described in Section 4.3 can be used. Also the term $\mathbf{Dz}(s)$ used for modeling the residual variability was shown to be unimportant with respect to the performance of the model [Burget et al., 2009].

As we are not going to deeply analyze the JFA in this work, we will skip the derivations of scoring and training and refer the reader to [Glembek, 2012, Kockmann, 2012] where a similar notation is used.

Chapter 5

i–vector Approach

The systems based on i–vectors have become the state-of-the-art technique in the speaker verification field [Dehak et al., 2010b]. I–vectors are also widely used in language recognition [Martínez et al., 2011]. Their origin is connected with the JFA technique and the summer 2008 Johns Hopkins University workshop on Robust Speaker Recognition [Burget et al., 2008]. At that time, JFA was the state-of-the-art technique in speaker verification and naturally, it was further investigated during the workshop. One of the research directions was to use the speaker factors from JFA as low-dimensional features for an SVM classifier. The results were surprisingly good and in addition, it was discovered that replacing the speaker factors by channel factors still yielded a system with a reasonable performance (around 20% EER). This finding revealed that, contrary to expectations, channel factors still contain a fair amount of speaker information. This was verified by a successful fusion of the two systems using speaker- and channel-factors. Najim Dehak then proposed to simplify the JFA model to a feature extractor with a single subspace that would contain all variability ("total variability"). He initially called the hidden variable a t–vector, but soon the community adopted the name "i–vector", where the "i" is left for interpretation.

Thanks to its simplicity, the method for i–vector extraction remained the same for the research systems, however, the success in the production environment inspired several approaches to its simplification and optimization [Glembek et al., 2011, Cumani and Laface, 2013, Cumani and Laface, 2014a].

5.1 Theoretical Background

The main idea behind the i–vector model is to transform the large utterance specific GMM supervector $\mathbf{s}$ into a small subspace, while retaining most of the important variability. From the perspective of speaker recognition, the supervector $\mathbf{s}$ contains both the speaker and inter-session characteristics of a given speech segment and is modeled according to:

$$\mathbf{s} = \mathbf{u} + \mathbf{T}\mathbf{w}, \tag{5.1}$$

where $\mathbf{u}$ is the UBM GMM mean supervector, composed of C GMM components of dimension F. $\mathbf{T}$ is a low-rank rectangular matrix representing M bases spanning the sub–space including important inter and intra–speaker variability in the supervector space. The subspace defined by the matrix $\mathbf{T}$ is often referred to as "i–vector subspace" or "total variability subspace". Vector $\mathbf{w}$ is a realization of a latent variable $\mathbf{W}$, of size M, having a standard normal prior distribution

$$\mathbf{W} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{5.2}$$

We can notice, that formally, this model is almost equivalent to the eigenvoice model described in Section 4.3.2. The difference with respect to eigenvoice model resides in tying the latent variable to every utterance, independent of speaker. The same steps as already described for the subspace modeling in Section 4.3 will apply also to i–vectors.

Ultimately, the aim is to estimate the parameters of the posterior distribution of the latent variable $\mathbf{W}$ for each set of τ input features extracted from the given speech segment $\mathcal{X} = \{\mathbf{x}_1\mathbf{x}_2 \dots \mathbf{x}_\tau\}$. Assuming the standard normal prior for $\mathbf{W}$, the posterior distribution is also Gaussian:

$$\mathbf{W}|\mathcal{X} \sim \mathcal{N}(\boldsymbol{\phi}_\mathcal{X}, \boldsymbol{\Gamma}_\mathcal{X}^{-1}), \tag{5.3}$$

with mean vector and precision matrix as in (4.33 and 4.34):

$$\begin{aligned} \boldsymbol{\phi}_\mathcal{X} &= \boldsymbol{\Gamma}_\mathcal{X}^{-1}\mathbf{T}^\mathrm{T}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}_\mathcal{X} \\ \boldsymbol{\Gamma}_\mathcal{X} &= \mathbf{I} + \sum_{c=1}^{C} N_\mathcal{X}^{(c)}\mathbf{T}^{(c)^\mathrm{T}}\boldsymbol{\Sigma}^{(c)^{-1}}\mathbf{T}^{(c)}, \end{aligned} \tag{5.4}$$

respectively. As in Chapter 4, in these equations, $N_\mathcal{X}^{(c)}$ (4.6) are the zero–order statistics collected with the UBM for the set of feature vectors in $\mathcal{X}$. $\mathbf{T}^{(c)}$ is the $F \times M$ sub-matrix of $\mathbf{T}$ corresponding to the c–th mixture component such that $\mathbf{T} = \left(\mathbf{T}^{(1)\mathrm{T}} \dots \mathbf{T}^{(C)\mathrm{T}}\right)^\mathrm{T}$, and $\tilde{\mathbf{f}}_\mathcal{X}$ is the supervector stacking the first–order statistics $\tilde{\mathbf{f}}_\mathcal{X}^{(c)}$, centered (see (4.9)) around the corresponding UBM means, $\boldsymbol{\Sigma}^{(c)}$ is the UBM c–th covariance matrix, $\boldsymbol{\Sigma}$ is a block diagonal matrix composed of matrices $\boldsymbol{\Sigma}^{(c)}$, and $\gamma_t^{(c)}$ is the occupation probability of feature vector $\mathbf{x}_t$ for the c-th Gaussian component.

The i-vector $\boldsymbol{\phi}$ – a low dimensional fixed-length vector, which represents the segment $\mathcal{X}$ of a variable length, is then computed as the MAP point estimate of the variable $\mathbf{W}$, i.e., the mean of the posterior distribution $P_{\mathbf{W}|\mathcal{X}}(\mathbf{w})$.

A Maximum-Likelihood estimate of matrix $\mathbf{T}$ can be obtained by following the steps from Section 4.3.1. Each submatrix $\mathbf{T}^c$ can be re-estimated as in (4.45):

$$\mathbf{T}^{(c)} = \sum_\mathcal{X} \left[\tilde{\mathbf{f}}_\mathcal{X}^{(c)}\boldsymbol{\phi}_\mathcal{X}(N_\mathcal{X}^{(c)}(\boldsymbol{\phi}_\mathcal{X}\boldsymbol{\phi}_\mathcal{X}^\mathrm{T} + \boldsymbol{\Gamma}_\mathcal{X}^{-1})^{-1}\right]. \tag{5.5}$$

Note that the we do not require any speaker labels and the $\mathbf{T}$ matrix is trained in an unsupervised way. The GMM subspace framework is then used as a feature extractor of the low-dimensional vectors containing most of the relevant variability from the original data – both useful and harmful for the target classification task. The presence of the unwanted variability in the i–vectors has to be dealt with when using i–vectors as features for classifiers or when using i–vectors directly for scoring.

5.2 Linear Discriminant Analysis

As stated above, using the i–vectors as features requires us to perform a compensation for the unwanted variability. Shortly after i–vectors were introduced, this problem was tackled by means of a simple Linear Discriminant Analysis (LDA) followed by Within-Class Covariance Normalization (WCCN) [Hatch et al., 2006, Dehak et al., 2009, Dehak et al., 2010a]. The LDA can be used to further reduce the dimensionality of the i–vectors and remove the dimensions corresponding to high within-class (intra-speaker) variability caused by the channel effects present in the original data. The LDA principles are also the basis of the Probabilistic Linear Discriminant Analysis (PLDA), nowadays considered to be one of the state-of-the-art techniques used for speaker recognition.

The LDA is based on maximizing Fisher's discriminant ratio – a ratio between across-class and within-class variability. Formally, when we are given a set of D-dimensional patterns $\mathbf{x}_i$ belonging to K classes C_k, we are searching for a linear projection

$$\mathbf{y}_i = \mathbf{A}\mathbf{x}_i, \tag{5.6}$$

where $\mathbf{A}$ is a $D' \times D$ transformation matrix with $D' \leq K - 1$, which maximizes the Fisher's criterion [Bishop, 2006]:

$$J(\mathbf{A}) = \mathrm{tr}\left\{(\mathbf{A}\boldsymbol{\Sigma}_{\mathrm{ac}}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\boldsymbol{\Sigma}_{\mathrm{ac}}\mathbf{A}^{\mathrm{T}})\right\}, \tag{5.7}$$

where $\boldsymbol{\Sigma}_{\mathrm{wc}}$ and $\boldsymbol{\Sigma}_{\mathrm{ac}}$ are within-class and across-class covariance matrices, respectively, which can be ML-estimated from training data as:

$$\boldsymbol{\Sigma}_{\mathrm{wc}} = \sum_{k=1}^{K} \sum_{n \in C_k} (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \tag{5.8}$$

$$\boldsymbol{\Sigma}_{\mathrm{ac}} = \sum_{k=1}^{K} N_k(\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^{\mathrm{T}} \tag{5.9}$$

with

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n \in C_k} \mathbf{x}_n \tag{5.10}$$

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{k=1}^{K} N_k \boldsymbol{\mu}_k, \tag{5.11}$$

where N_k is the number of patters for class k and N is the total number of patterns. The solution for $\mathbf{A}$ is then given by D' eigen vectors corresponding to the largest eigenvalues of $\boldsymbol{\Sigma}_{\mathrm{wc}}^{-1}\boldsymbol{\Sigma}_{ac}$. In the Figure 5.1, we show the meaning of LDA parameters with relation to speaker recognition.

5.2.1 Within Class Covariance Normalization

Within-Class Covariance Normalization was first applied in speaker recognition in [Hatch et al., 2006, Dehak et al., 2009] to SVM-based one-versus-all model for speaker

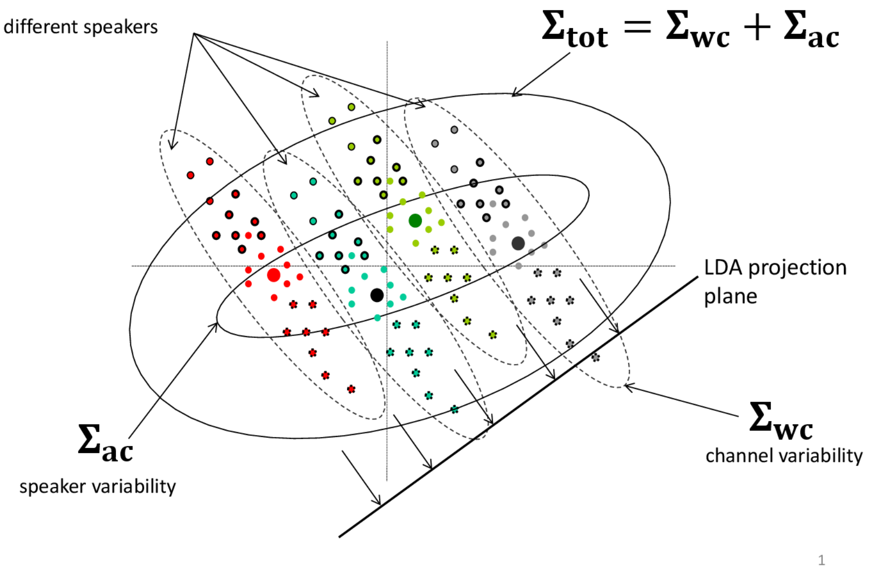


Figure 5.1: Demonstration of the LDA (and also PLDA) assumptions about the data: The bold points correspond to the speaker means in the i–vector space. The conditional distributions of the i–vectors around the speaker means share a common within-class covariance $\mathbf{\Sigma}_{\mathrm{wc}}$. The distribution of all speaker means is then depicted as the Gaussian with the across-class covariance matrix $\mathbf{\Sigma}_{\mathrm{ac}}$. For completeness, the total covariance of the data is shown as a sum of the two covariances $\mathbf{\Sigma}_{\mathrm{wc}} = \mathbf{\Sigma}_{\mathrm{wc}} + \mathbf{\Sigma}_{\mathrm{ac}}$.

recognition. WCCN is again a linear transformation:

$$\mathbf{z}_{\mathrm{norm}} = \mathbf{B}\mathbf{z}, \tag{5.12}$$

where $\mathbf{B}$ is a square transformation matrix. We search for such $\mathbf{B}$ that after applying it to the data $\mathbf{z}$, the within-class covariance matrix $\mathbf{\Sigma}_{\mathrm{wc}}$ as defined in (5.8) becomes identity. By such transformation, we scale the directions in the i–vector space inversely proportional to an estimate of the within class covariance and effectively reduce the unwanted channel variability. The solution for this problem can be obtained by Cholesky decomposition of $\mathbf{B}\mathbf{B}^{\mathrm{T}} = \mathbf{\Sigma}_{\mathrm{wc}}^{-1}$:

$$\mathbf{B} = \mathrm{chol}(\mathbf{\Sigma}_{\mathrm{wc}}^{-1}). \tag{5.13}$$

5.3 Cosine Distance Scoring

Before venturing into describing different variants of PLDA, which model the i–vector generation process, we describe a simple technique using i–vectors directly for obtaining the verification score. Najim Dehak originally proposed to classify i–vectors using Support Vector Machines [Dehak et al., 2010b], which lead to introduction of a simple cosine distance scoring metric that is measuring the angle between two i–vectors:

$$s(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \frac{\boldsymbol{\phi}_1^{\mathrm{T}} \boldsymbol{\phi}_2}{\|\boldsymbol{\phi}_1\| \|\boldsymbol{\phi}_1\|}. \tag{5.14}$$

This metric was originally used as a SVM kernel, but later, it turned out to be a good verification score on its own. It should be noted, that the scores obtained this way are symmetric to swapping enroll and test i–vector.

To perform the inter-session compensation, we incorporate already discussed LDA and WCCN into the scoring metric and obtain

$$s(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \frac{(\mathbf{A}^{\mathrm{T}}\boldsymbol{\phi}_1)^{\mathrm{T}}\mathbf{B}(\mathbf{A}^{\mathrm{T}}\boldsymbol{\phi}_2)}{\sqrt{(\mathbf{A}^{\mathrm{T}}\boldsymbol{\phi}_1)^{\mathrm{T}}\mathbf{B}(\mathbf{A}^{\mathrm{T}}\boldsymbol{\phi}_1)}\sqrt{(\mathbf{A}^{\mathrm{T}}\boldsymbol{\phi}_2)^{\mathrm{T}}\mathbf{B}(\mathbf{A}^{\mathrm{T}}\boldsymbol{\phi}_2)}}. \tag{5.15}$$

It should be also noted that the scores obtained simply as a distance of two vectors can not be interpreted as a log-likelihood ratios. If the output in form of the log-likelihood ratios is requested, the scores have to be calibrated [Brümmer, 2010b]. To achieve a good performance with this approach, a score normalization has to be applied. Since the scoring is very efficient, standard techniques like ZT-norm (see Section 1.2.1) can be performed very efficiently. We can simply add the set of i–vectors representing the T-norm utterances to the set of enrollment i–vectors and a set of Z-norm i–vectors to the test i–vectors and compute the whole matrix of scores. This matrix will already include all scores needed to compute the ZT-norm statistics.

Chapter 6

Probabilistic Linear Discriminant Analysis

In the last four years, SRE systems based on the i–vectors and Probabilistic Linear Discriminant Analysis (PLDA) became state-of-the-art. In PLDA model, an i–vector $\boldsymbol{\phi}$ is considered to be a realization of a random variable $\boldsymbol{\Phi}$, whose generation process can be described in terms of a set of latent variables. Different PLDA models exist, which use different numbers of hidden variables as well as different priors. The two favorite models are heavy-tailed PLDA (HTPLDA) [Kenny, 2010], where Student's t-distribution is imposed on the latent variables and the PLDA [Prince and Elder, 2007], which assumes Gaussian priors.

All PLDA models for speaker recognition [Kenny, 2010] and [Brümmer and de Villiers, 2010], however, represent the speaker identity in terms of a latent variable $\mathbf{Y}$ which is assumed to be tied across all segments of the same speaker. Usually, inter–speaker variability for a speech segment $\mathcal{X}_i$ is represented by hidden variable $\mathbf{X}_i$. The hidden variables $\mathbf{X}_i$ are assumed to be i.i.d. with respect to the speech segments.

In the most common PLDA model, an i–vector $\boldsymbol{\phi}$ is the sum of multiple terms [Kenny, 2010]:

$$\boldsymbol{\phi} = \mathbf{m} + \mathbf{U}\mathbf{y} + \mathbf{V}\mathbf{x} + \mathbf{e}, \tag{6.1}$$

where $\mathbf{m}$ is the i–vector mean, $\mathbf{y}$ is a realization of the speaker identity variable $\mathbf{Y}$, $\mathbf{x}$ is the realization of channel variable $\mathbf{X}$ and $\mathbf{e}$ is the realization of the residual noise $\mathbf{E}$.

The role of matrices $\mathbf{U}$ and $\mathbf{V}$ is to constrain the dimension of the subspaces for $\mathbf{y}$ and $\mathbf{x}$, providing the bases for a speaker subspace, often called "eigenvoices" and bases for a channel subspace, usually called "eigenchannels". In this work, we will assume standard normal priors for the speaker identity variable $\mathbf{Y}$ and channel variable $\mathbf{X}$. The noise $\mathbf{E}$ is assumed to be Gaussian distributed with the diagonal covariance matrix of the residual data variability $\mathbf{D}^{-1}$:

$$\mathbf{Y} \sim \mathcal{N}(0, \mathbf{I}) \tag{6.2}$$

$$\mathbf{X} \sim \mathcal{N}(0, \mathbf{I}) \tag{6.3}$$

$$\mathbf{E} \sim \mathcal{N}(0, \mathbf{D}^{-1}). \tag{6.4}$$

In case of this PLDA model, the across-class covariance matrix is defined as $\mathbf{\Sigma}_{\mathrm{ac}} = \mathbf{U}^{\mathrm{T}}\mathbf{U}$, which is often low rank and limits the speaker variability to live in a subspace spanned by the columns of the reduced rank matrix $\mathbf{U}$. Similarly, the within-class covariance matrix is defined as $\mathbf{\Sigma}_{\mathrm{ac}} = \mathbf{V}^{\mathrm{T}}\mathbf{V} + \mathbf{D}^{-1}$.

A more complex model assuming the Student's t-distribution of the priors is computationally more expensive, as it does not have a closed form solution for computing posteriors and therefore the time required both for training and testing is significantly increased.

The advantage of imposing more relaxed heavy-tailed priors lies in the ability of the model to cope with the raw i–vectors without previous normalization. It was shown that preconditioning the i–vectors by means of length normalization [Garcia-Romero, 2011] allows the Gaussian PLDA to yield results comparable to HTPLDA. For the discussion on the normalization of the i–vectors, see Chapter 8.

In the HTPLDA model, we would introduce scalar parameters n_1, n_2 and ν, referred to as degrees of freedom, and scalar hidden variables u_1, u_2 and v_r. Then we would assume that

$$\mathbf{Y} \sim \mathcal{N}(0, u_1^{-1}\mathbf{I}), u_1 \sim \mathcal{G}(n_1/2, n_1/2) \tag{6.5}$$

$$\mathbf{X} \sim \mathcal{N}(0, u_2^{-1}\mathbf{I}), u_2 \sim \mathcal{G}(n_2/2, n_2/2) \tag{6.6}$$

$$\mathbf{E} \sim \mathcal{N}(0, v_r^{-1}\mathbf{D}^{-1}), v_r \sim \mathcal{G}(\nu/2, \nu/2), \tag{6.7}$$

where $\mathcal{G}(a, b)$ represents a Gamma distribution with parameters a and b.

Although, we will provide some experimental results with HTPLDA, we will concentrate on the variants of Gaussian PLDA.

6.1 Trial Scoring

Given the sets of enrollment and test segments forming a speaker verification trial, we obtain a speaker verification score. In this section, we will define the score as a log-likelihood ratio between the hypotheses that all of the segments were generated by the same speaker and that each set of segments was generated independently by a different speaker.

Since i–vectors are assumed independent given the hidden variables, the likelihood that a set of n speech segments $\mathcal{X}_1 \ldots \mathcal{X}_n$ belongs to the same speaker (hypothesis H_s) can be evaluated as:

$$\begin{aligned} l\left(\mathcal{X}_1 \ldots \mathcal{X}_n | H_s\right) &= P_{\mathbf{\Phi}_1 \ldots \mathbf{\Phi}_n}(\boldsymbol{\phi}_1 \ldots \boldsymbol{\phi}_n | H_s) \\ &= \int_{\mathbf{y}} \int_{\mathbf{x}_1} \cdots \int_{\mathbf{x}_n} \prod_{i=1}^{n} \left[P_{\mathbf{\Phi}_i|\mathbf{Y},\mathbf{X}_i}\left(\boldsymbol{\phi}_i | \mathbf{y}, \mathbf{x}_i\right) P_{\mathbf{X}_i}\left(\mathbf{x}_i\right) d\mathbf{x}_i \right] \cdot P_{\mathbf{Y}}(\mathbf{y}) \mathrm{d}\mathbf{y}, \end{aligned} \tag{6.8}$$

where $\boldsymbol{\phi}_i$ is the i–vector extracted from segment $\mathcal{X}_i$, $P_{\mathbf{\Phi}_1 \ldots \mathbf{\Phi}_n | H_s}(\boldsymbol{\phi}_1 \ldots \boldsymbol{\phi}_n)$ is the joint probability of the i–vectors given the same speaker hypothesis H_s, $P_{\mathbf{X}}(\mathbf{x})$ and $P_{\mathbf{Y}}(\mathbf{y})$

are the prior distributions for $\mathbf{X}$ and $\mathbf{Y}$, respectively. $P_{\mathbf{\Phi}|\mathbf{Y},\mathbf{X}}\left(\boldsymbol{\phi}|\mathbf{y},\mathbf{x}\right)$ is the conditional distribution of an i–vector given the hidden variables. It is related to the distribution $P_{\mathbf{E}}(\mathbf{e})$ of the noise term by $P_{\mathbf{\Phi}|\mathbf{Y},\mathbf{X}}\left(\boldsymbol{\phi}|\mathbf{y},\mathbf{x}\right) = P_{\mathbf{E}}(\boldsymbol{\phi} - \mathbf{m} - \mathbf{U}\mathbf{y} - \mathbf{V}\mathbf{x})$.

In order to obtain an inference about the speaker identity, we ask the question, whether a set of n enrollment segments $\mathcal{X}_{e_1}\ldots\mathcal{X}_{e_n}$ for a known (target) speaker and a set of m test segments of a single unknown speaker $\mathcal{X}_{t_1}\ldots\mathcal{X}_{t_m}$ belong to the same speaker or not. Specifically, we want to compute the log-likelihood ratio of the segments being observed under the same speaker and different speaker hypotheses

$$s = \log\frac{l\left(\mathcal{X}_{e_1}\ldots\mathcal{X}_{e_n},\mathcal{X}_{t_1}\ldots\mathcal{X}_{t_m}|H_s\right)}{l\left(\mathcal{X}_{e_1}\ldots\mathcal{X}_{e_n},\mathcal{X}_{t_1}\ldots\mathcal{X}_{t_m}|H_d\right)}. \tag{6.9}$$

Since speaker factors are assumed independent, the speaker verification log–likelihood ratio s can be formulated as:

$$s = \log\frac{l\left(\mathcal{X}_{e_1}\ldots\mathcal{X}_{e_n},\mathcal{X}_{t_1}\ldots\mathcal{X}_{t_m}|H_s\right)}{l\left(\mathcal{X}_{e_1}\ldots\mathcal{X}_{e_n}|H_s\right)l\left(\mathcal{X}_{t_1}\ldots\mathcal{X}_{t_m}|H_s\right)}. \tag{6.10}$$

It is worth noting, that the log-likelihood ratio calculated in this way is symmetric in terms of swapping the enroll and test sets. Also note that standard i–vector, which is extracted by MAP point estimate of the posterior distribution of $\mathbf{W}$ given $\mathcal{X}$, and classified by PLDA, does not embed the intrinsic uncertainty of its estimate. We will address this fact in the next chapter, where we will extend the PLDA model and no longer consider the segment $\mathcal{X}$ being represented by a single i–vector, by the i–vector distribution $\mathbf{W}|\mathcal{X}$.

6.2 Simplified PLDA Model

It is convenient to assume that the noise term $\mathbf{E}$ has a full covariance matrix, so that the terms $\mathbf{V}\mathbf{x}$ and $\mathbf{e}$ in (6.1) can be merged. Therefore, in our approach, a distribution of i–vector $\boldsymbol{\phi}$ is modeled as:

$$\boldsymbol{\phi} = \mathbf{m} + \mathbf{U}\mathbf{y} + \mathbf{e}. \tag{6.11}$$

In this model, we restrict only the speaker variability to reside in the subspace spanned by the reduced rank matrix $\mathbf{U}$. The across-class covariance matrix is again defined as $\mathbf{\Sigma}_{ac} = \mathbf{U}^{\mathrm{T}}\mathbf{U}$. Channel variability is then modeled by a full rank within-class covariance matrix $\mathbf{\Sigma}_{wc} = \mathbf{\Lambda}^{-1}$. Speaker factors and the residual noise priors are assumed to be Gaussian, i.e.:

$$\mathbf{Y} \sim \mathcal{N}(0,\mathbf{I})\ , \quad \mathbf{E} \sim \mathcal{N}(0,\mathbf{\Lambda}^{-1})\ , \tag{6.12}$$

where $\mathbf{\Lambda}$ is the precision matrix of noise $\mathbf{E}$. According to (6.11) and (6.12), the conditional distribution of an i–vector random variable $\mathbf{\Phi}$ given a value $\mathbf{y}$ for the speaker identity $\mathbf{Y}$ is:

$$\mathbf{\Phi}|\left(\mathbf{Y} = \mathbf{y}\right) \sim \mathcal{N}(\mathbf{m} + \mathbf{U}\mathbf{y},\mathbf{\Lambda}^{-1}). \tag{6.13}$$

Omitting the channel factors, which are now embedded in the noise term, the likelihood that the n speech segments $\mathcal{X}_1\ldots\mathcal{X}_n$ belong to the same speaker can be computed by means of a simplified expression of (6.8) as:

$$l(\mathcal{X}_1\ldots\mathcal{X}_n|H_s) = P_{\mathbf{\Phi}_1\ldots\mathbf{\Phi}_n}(\boldsymbol{\phi}_1\ldots\boldsymbol{\phi}_n|H_s)$$

$$= \int_{\mathbf{y}} \prod_{i=1}^{n} P_{\mathbf{\Phi}_i|\mathbf{Y}}(\boldsymbol{\phi}_i|\mathbf{y}) P_{\mathbf{Y}}(\mathbf{y}) \mathrm{d}\mathbf{y}. \tag{6.14}$$

6.2.1 Closed-Form Solution for Scoring

In order to compute the likelihood of a set of n i–vectors $\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n$ (or corresponding speech segments $\mathcal{X}_1 \dots \mathcal{X}_n$), we observe that the joint log-likelihood of the i–vectors and the hidden variables is:

$$\begin{aligned} \log P_{\mathbf{\Phi}_1 \dots \mathbf{\Phi}_n, \mathbf{Y}}(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n, \mathbf{y}|H_s) &= \sum_{i=1}^{n} \log P_{\mathbf{\Phi}|\mathbf{Y}}(\boldsymbol{\phi}_i|\mathbf{y}) + \log P_{\mathbf{Y}}(\mathbf{y}) \\ = \sum_{i=1}^{n} \left[-\frac{1}{2} (\boldsymbol{\phi}_i - \mathbf{m} - \mathbf{U}\mathbf{y})^{\mathrm{T}} \mathbf{\Lambda} \left(\boldsymbol{\phi}_i - \mathbf{m} - \mathbf{U}\mathbf{y} \right) \right] &+ \frac{1}{2} \mathbf{y}^{\mathrm{T}} \mathbf{y} + k, \end{aligned} \tag{6.15}$$

where k is a constant collecting the terms that do not depend on speaker identity $\mathbf{y}$. Since equation (6.15) is a quadratic function, using "completion of squares", we can observe that the posterior distribution of $\mathbf{Y}$ given a set of i–vectors is Gaussian

$$\mathbf{Y}|\mathbf{\Phi}_1 \dots \mathbf{\Phi}_n \sim \mathcal{N}(\hat{\mathbf{y}}, \mathbf{P}^{-1}), \tag{6.16}$$

with precision matrix and mean:

$$\begin{aligned} \mathbf{P} &= \mathbf{I} + \mathbf{U}^{\mathrm{T}} \mathbf{\Lambda} \mathbf{U} \\ \hat{\mathbf{y}} &= \mathbf{P}^{-1} \mathbf{U}^{\mathrm{T}} \sum_{i=1}^{n} \mathbf{\Lambda} \left(\boldsymbol{\phi}_i - \mathbf{m} \right). \end{aligned} \tag{6.17}$$

The likelihood that a set of segments belongs to the same speaker can be written as:

$$P_{\mathbf{\Phi}_1 \dots \mathbf{\Phi}_n}(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n|H_s) = \frac{P(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n|\mathbf{y}_0) P(\mathbf{y}_0)}{P(\mathbf{y}_0|\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n)}, \tag{6.18}$$

where $\mathbf{y}_0$ is an arbitrary vector, which does not cause the denominator to be zero. For the convenience, we can set the $\mathbf{y}_0 = \mathbf{0}$, so that $\mathbf{U}\mathbf{y}_0 = 0$ and derive a closed form solution for the same speaker hypothesis [Brümmer and de Villiers, 2010]:

$$\begin{aligned} \log P_{\mathbf{\Phi}_1 \dots \mathbf{\Phi}_n}(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n|H_s) = \sum_{i=1}^{n} &\left[\frac{1}{2} \log |\mathbf{\Lambda}| - \frac{M}{2} \log 2\pi - \frac{1}{2} (\boldsymbol{\phi}_i - \mathbf{m})^{\mathrm{T}} \mathbf{\Lambda} (\boldsymbol{\phi}_i - \mathbf{m}) \right] \\ &- \frac{1}{2} \log |\mathbf{P}| + \frac{1}{2} \hat{\mathbf{y}}^{\mathrm{T}} \mathbf{P} \hat{\mathbf{y}} - \frac{S}{2} \log 2\pi, \end{aligned} \tag{6.19}$$

where M is the i–vector dimension, and S is the speaker factor dimension.

6.2.2 Training of PLDA Model Parameters

Following [Brümmer, 2010a], we will derive an update of the simplified PLDA parameters using EM algorithm with the minimum divergence step. We have already defined a global i–vector mean $\mathbf{m}$. Our training data consist of $s = 1 \dots |\mathcal{Y}|$ speakers, where $\mathcal{Y}$ is a set of all speakers, where for every speaker s, we have $i = 1 \dots n_s$ observations. Every observation is represented as an i–vector $\boldsymbol{\phi}_{s,i}$. The zero order statistics per speaker are simply numbers of observations per speaker n_s and the first order statistics centered around the mean $\mathbf{m}$ are defined as:

$$\mathbf{f}_s = \sum_{i=1}^{n_s} \boldsymbol{\phi}_{s,i} - \mathbf{m}. \tag{6.20}$$

The global zero- and second-order statistic are given by:

$$N = \sum_{s \in \mathcal{Y}} n_s, \tag{6.21}$$

$$\mathbf{S} = \sum_{s \in \mathcal{Y}} \sum_{i=1}^{n_s} (\boldsymbol{\phi}_{s,i} - \mathbf{m})(\boldsymbol{\phi}_{s,i} - \mathbf{m})^{\mathrm{T}}. \tag{6.22}$$

Using the precision matrix $\mathbf{P}$ and mean $\hat{\mathbf{y}}$ from (6.17) of the posterior distribution of speaker factors $\mathbf{y}$ (6.16), we will also define auxiliary statistics $\mathbf{Q}$ and $\mathbf{R}$ accumulated over all speakers:

$$\mathbf{Q} = \sum_{s \in \mathcal{Y}} \hat{\mathbf{y}}_s \mathbf{f}_s^{\mathrm{T}}, \tag{6.23}$$

$$\mathbf{R} = \sum_{s \in \mathcal{Y}} n_s (\mathbf{P}_s^{-1} + \mathbf{y}_s \mathbf{y}_s^{\mathrm{T}}). \tag{6.24}$$

The data likelihood for a speaker s is given by

$$\begin{aligned} P_{\boldsymbol{\Phi}_1 \dots \boldsymbol{\Phi}_{n_s} | \mathbf{Y} = \mathbf{y}_s}(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_{n_s}, \mathbf{U}, \boldsymbol{\Lambda}) &= \prod_{i=1}^{n_s} \mathcal{N}(\boldsymbol{\phi}_{s,i} | \mathbf{U}\mathbf{y}_s, \boldsymbol{\Lambda}^{-1}) \\ &= \exp\left(-\frac{NM}{2} \log(2\pi) - \frac{N}{2} \log|\Lambda^{-1}| + \right. \\ &\left. + \sum_{i=1}^{n_s} (-\frac{1}{2} \boldsymbol{\phi}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \boldsymbol{\phi}_{s,i} + \boldsymbol{\phi}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U} \mathbf{y}_s) - \frac{1}{2} \mathbf{y}_s^{\mathrm{T}} \mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U} \mathbf{y}_s \right), \end{aligned} \tag{6.25}$$

and by omitting the terms not dependent on $\mathbf{U}$ and $\boldsymbol{\Lambda}$, we get the EM auxiliary function to maximize:

$$\begin{aligned} &\sum_s \langle \log P_{\boldsymbol{\Phi}_1 \dots \boldsymbol{\Phi}_{n_s} | \mathbf{Y} = \mathbf{y}_s}(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_{n_s}, \mathbf{U}, \boldsymbol{\Lambda}) \rangle_{P(\mathbf{y}_s | \boldsymbol{\phi}_{s,1} \dots \boldsymbol{\phi}_{s,n_s})} \\ &\quad = \frac{N}{2} \log|\boldsymbol{\Lambda}| - \frac{1}{2} \mathrm{tr}(\mathbf{S}\boldsymbol{\Lambda}) - \frac{1}{2} \mathrm{tr}(\mathbf{R}\mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U}) + \mathrm{tr}(\mathbf{Q}\boldsymbol{\Lambda}\mathbf{U}) + \mathrm{const}. \end{aligned} \tag{6.26}$$

We obtain the maximum likelihood update for $\mathbf{U}$ and $\mathbf{\Lambda}^{-1}$ by taking the derivative of EM auxiliary function w.r.t. corresponding parameters and setting them to zero:

$$\begin{aligned} \mathbf{U} &= \mathbf{Q}^{\mathrm{T}}\mathbf{R}^{-1} && (6.27) \\ \mathbf{\Lambda}^{-1} &= \frac{1}{N}(\mathbf{S} - \mathbf{U}\mathbf{Q} - \mathbf{Q}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} + \mathbf{U}\mathbf{R}\mathbf{U}^{\mathrm{T}}) \\ &= \frac{1}{N}(\mathbf{S} - \mathbf{U}\mathbf{Q}). && (6.28) \end{aligned}$$

For faster convergence, we can apply minimal divergence update [Brümmer, 2009, Kenny, 2005], and force the speaker factors to be standard normal distributed. We can achieve this goal by rotating the maximum-likelihood estimate of $\mathbf{U}$. For this purpose, we accumulate

$$\mathbf{A} = \frac{1}{|\mathcal{Y}|}\sum_{s \in \mathcal{Y}}(\mathbf{P}_s^{-1} + \mathbf{y}_s\mathbf{y}_s^{\mathrm{T}}). \tag{6.29}$$

And finally, a minimum divergence re-estimation of $\mathbf{U}$ is:

$$\mathbf{U} \leftarrow \mathbf{U}\,\mathrm{chol}(\mathbf{A})^{\mathrm{T}}, \tag{6.30}$$

where $\mathrm{chol}(\mathbf{M})\,\mathrm{chol}(\mathbf{M})^{\mathrm{T}} = \mathbf{M}$ denotes Cholesky decomposition.

This scheme is used in an iterative way until convergence. The number of iterations is usually small and depends mainly on the size of dataset, number of speakers and size of the subspace. Usually ten iterations is enough, with fifty being the safe upper limit for most scenarios.

Chapter 7

Full Posterior Distribution PLDA Model

In this chapter, we will demonstrate, how to extend the standard PLDA model, where we considered the utterance to be sufficiently well represented by a single i–vector. We will show that the simple and effective PLDA framework can still be used even if a speech segment is no more represented by a single i–vector but by its posterior distribution. In particular, we will derive the formulation of likelihood for a standard Gaussian PLDA model based on the i–vector posterior distribution, and propose a new PLDA model where the inter–speaker variability is assumed to have an utterance–dependent distribution. We will show that it is possible to rely on the standard PLDA framework simply replacing the PLDA likelihood definition.

It is well known, that the goodness of the i–vector estimate depends mainly on the covariance of the distribution, which accounts for the "uncertainty" of the i–vector extraction process. This ucertainity of the i–vector estimate is however not exploited by many standard and popular classifiers based on i–vectors, such as the ones based on cosine distance scoring [Dehak et al., 2010b], PLDA [Kenny, 2010], discriminative PLDA [Burget et al., 2011] or SVMs [Cumani et al., 2013].

The i–vector covariance depends on the zero–order statistics estimated using a UBM for the set of observed features (see equation (5.4) in Chapter 5). These statistics are affected by several factors such as the noise level, the channel characteristics, and the acoustic content of the observed features, but the predominant factor is the number of the observed feature frames – *duration of a given utterance*. Shorter utterances tend to produce larger covariances, so that i–vector estimates become less reliable.

7.1 Incorporating the I–vector Posterior Distribution into PLDA

The standard i–vector, which is extracted by MAP point estimate of the posterior distribution of $\mathbf{W}$ given $\mathcal{X}$ (see (5.3)) does not embed the intrinsic uncertainty of its estimate. Remembering the likelihood computation for the standard PLDA (see 6.8), we can extend this model by considering all possible i–vectors, which correspond to the speech segments

$\mathcal{X}_1 \dots \mathcal{X}_n$.

We refer to this new model as the PLDA based on the Full Posterior Distribution (FPD–PLDA) of $\mathbf{W}$ given $\mathcal{X}$. As previously mentioned, we now assume that every segment $\mathcal{X}$ is no longer represented by a single i–vector corresponding to the most likely value of the latent variable $\mathbf{w}$ in the i–vector model (5.1). Instead, segment $\mathcal{X}$ will be represented by the i–vector extractor distribution $\mathbf{W}|\mathcal{X}$ (see (5.3)). Therefore, the uncertainty in i–vector estimate will be taken into account. In the following text, we will refer to the posterior distribution $\mathbf{W}|\mathcal{X}$ simply as to i–vector posterior distribution.

The PLDA model allows computing the likelihood of a speech segment given a realization $\mathbf{w}$ of the random variable $\mathbf{W}|\mathcal{X}$. The likelihood of a set of segments $\mathcal{X}_1 \dots \mathcal{X}_n$, thus, can be evaluated by integrating the PLDA likelihood (see equations (6.8) and (6.15)) over all possible realizations following the posterior distribution $\mathbf{W}|\mathcal{X}_1 \dots \mathcal{X}_n$.

$$l\left(\mathcal{X}_1 \dots \mathcal{X}_n | H_s\right) = \int_{\mathbf{w}_1} \cdots \int_{\mathbf{w}_n} P_{\mathbf{W}_1 \dots \mathbf{W}_n}\left(\mathbf{w}_1 \dots \mathbf{w}_n \middle| H_s\right) \prod_{i=1}^{n} \left[P_{\mathbf{W}_i|\mathcal{X}_i}(\mathbf{w}_i) \mathrm{d}\mathbf{w}_i \right], \tag{7.1}$$

where the first factor is the likelihood of the segments according to the original PLDA model given realizations $\mathbf{w}_1, \dots, \mathbf{w}_n$ of the i–vector posterior random variables, computed as in (6.8), and the second factor is the posterior probability of realizations $\mathbf{w}_1, \dots, \mathbf{w}_n$ representing segments $\mathcal{X}_1 \dots \mathcal{X}_n$ according to the i–vector extractor model. Using the form of (6.8) in (7.1), the likelihood can be rewritten as:

$$\begin{aligned} l\left(\mathcal{X}_1 \dots \mathcal{X}_n | H_s\right) = \int_{\mathbf{w}_1} \cdots \int_{\mathbf{w}_n} \int_{\mathbf{y}} \int_{\mathbf{x}_1} \cdots \int_{\mathbf{x}_n} \prod_{i=1}^{n} \Big[& P_{\mathbf{W}_i|\mathbf{Y},\mathbf{X}_i}\left(\mathbf{w}_i | \mathbf{y}, \mathbf{x}_i\right) \\ & \cdot P_{\mathbf{X}_i}(\mathbf{x}_i) P_{\mathbf{W}_i|\mathcal{X}_i}\left(\mathbf{w}_i\right) \mathrm{d}\mathbf{x}_i \mathrm{d}\mathbf{w}_i \Big] P_{\mathbf{Y}}(\mathbf{y}) \mathrm{d}\mathbf{y} \ . \end{aligned} \tag{7.2}$$

It is worth noting that, if the posterior for $\mathbf{W}|\mathcal{X}$ is replaced by a delta distribution centered in the posterior mean $\delta(\boldsymbol{\phi}_{\mathcal{X}})$, the likelihood of the original PLDA model using MAP–estimated i–vectors, given by (6.8), is obtained.

7.2 Extending the Classical Simplified PLDA

We will continue with the derivations using the simplified PLDA model introduced in previous Section 6.2. Starting from the point where we introduced the likelihood of a set of segments given the same-speaker hypothesis in (6.14), we introduce the full i–vector posterior into the equation and we get:

$$\begin{aligned} l\left(\mathcal{X}_1 \dots \mathcal{X}_n | H_s\right) &= \int_{\mathbf{w}_i} \cdots \int_{\mathbf{w}_n} \int_{\mathbf{y}} P_{\mathbf{Y}}(\mathbf{y}) \cdot \prod_{i=1}^{n} \left[P_{\mathbf{W}_i|\mathbf{Y}}(\mathbf{w}_i|\mathbf{y}) P_{\mathbf{W}_i|\mathcal{X}_i}(\mathbf{w}_i) \mathrm{d}\mathbf{w}_i \right] \mathrm{d}\mathbf{y} \\ &= \int_{\mathbf{y}} P_{\mathbf{Y}}(\mathbf{y}) \prod_{i=1}^{n} \left[\int_{\mathbf{w}_i} P_{\mathbf{W}_i|\mathbf{Y}}(\mathbf{w}_i|\mathbf{y}) P_{\mathbf{W}_i|\mathcal{X}_i}(\mathbf{w}_i) \mathrm{d}\mathbf{w}_i \right] \mathrm{d}\mathbf{y}. \end{aligned} \tag{7.3}$$

According to the Gaussian assumptions given in (5.3) and (6.12), the inner integral can be computed as

$$\begin{aligned}
&\int_{\mathbf{w}_i} P_{\mathbf{W}_i|\mathbf{Y}}(\mathbf{w}_i|\mathbf{y}) P_{\mathbf{W}_i|\mathcal{X}_i}(\mathbf{w}_i) \mathrm{d}\mathbf{w}_i = \\
&\int_{\mathbf{w}_i} \frac{1}{(2\pi)^{\frac{M}{2}} \left|\boldsymbol{\Lambda}^{-1}\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{w}_i - \mathbf{m} - \mathbf{U}\mathbf{y})^{\mathrm{T}} \boldsymbol{\Lambda} (\mathbf{w}_i - \mathbf{m} - \mathbf{U}\mathbf{y})} \\
&\cdot \frac{1}{(2\pi)^{\frac{M}{2}} \left|\boldsymbol{\Gamma}_i^{-1}\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{w}_i - \boldsymbol{\phi}_i)^{\mathrm{T}} \boldsymbol{\Gamma}_i (\mathbf{w}_i - \boldsymbol{\phi}_i)} \mathrm{d}\mathbf{w}_i,
\end{aligned} \tag{7.4}$$

where $\boldsymbol{\phi}_i$ and $\boldsymbol{\Gamma}_i$ are the mean and precision matrix of $\mathbf{W}_i|\mathcal{X}_i$ computed as in (5.4). Integral (7.4) can be interpreted as the convolution of two Gaussian distributions, leading to

$$\begin{aligned}
l(\mathcal{X}_1 \ldots \mathcal{X}_n|\mathbf{Y} = \mathbf{y}) = &\prod_{i=1}^{n} \frac{1}{(2\pi)^{\frac{M}{2}} \left|\boldsymbol{\Lambda}^{-1} + \boldsymbol{\Gamma}_i^{-1}\right|^{\frac{1}{2}}} \\
&\cdot e^{(\boldsymbol{\phi}_i - \mathbf{m} - \mathbf{U}\mathbf{y})^{\mathrm{T}} \left(\boldsymbol{\Lambda}^{-1} + \boldsymbol{\Gamma}_i^{-1}\right)^{-1} (\boldsymbol{\phi}_i - \mathbf{m} - \mathbf{U}\mathbf{y})}.
\end{aligned} \tag{7.5}$$

Comparing (7.5) and (6.15), we can see that now the covariance matrix of noise becomes segment-dependent as $[\boldsymbol{\Lambda}^{-1} + \boldsymbol{\Gamma}^{-1}]$. Considering the similarity of both models, we can say that the FPD-PLDA can be equivalently represented (likelihood calculation can be "simulated") by the standard PLDA modeling the usual i–vectors (i.e. i–vector posterior means), while assuming modified utterance dependent prior imposed on residual noise

$$\overline{\mathbf{E}}_i \sim \mathcal{N}\left(\mathbf{0}, \left[\boldsymbol{\Lambda}^{-1} + \boldsymbol{\Gamma}_i^{-1}\right]\right). \tag{7.6}$$

7.3 Scoring with FPD-PLDA

The log–likelihood that a set of segments belongs to the same speaker can be obtained by means of the same steps followed for the standard Gaussian PLDA model, just using the modified likelihood in (7.5). The new PLDA model can be described as:

$$\boldsymbol{\phi} = \mathbf{m} + \mathbf{U}\mathbf{y} + \overline{\mathbf{e}}\ , \tag{7.7}$$

as in (6.11), but with a segment–dependent distribution of the residual noise $\overline{\mathbf{E}}$. The i–vector associated to the speech segment $\mathcal{X}_i$ is again the mean $\boldsymbol{\phi}_i$ of the i–vector posterior $\mathbf{W}_i|\mathcal{X}_i$, but the priors of the PLDA parameters are given by:

$$\overline{\mathbf{E}}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}^{-1} + \boldsymbol{\Gamma}_i^{-1}) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}_{eq,i}^{-1})\ , \mathbf{Y} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})\ , \tag{7.8}$$

where

$$\boldsymbol{\Lambda}_{eq,i} = \left(\boldsymbol{\Lambda}^{-1} + \boldsymbol{\Gamma}_i^{-1}\right)^{-1}. \tag{7.9}$$

In the following text, to simplify the notation, we will refer to distributions without explicitly naming the corresponding hidden variable, e.g., we will write $P(\mathbf{y})$ rather than $P_{\mathbf{Y}}(\mathbf{y})$.

To compute the likelihood of a set of n i–vectors $\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n$ (i.e., of the set of speech segments $\mathcal{X}_1 \dots \mathcal{X}_n$), we follow the same steps as in the previous section 6.2.2 on the standard PLDA. Similarly to (6.15), we observe that the joint log–likelihood of the i–vectors and the hidden variables is:

$$\begin{aligned}\log P(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n, \mathbf{y}|H_s) &= \sum_{i=1}^{n} \log P(\boldsymbol{\phi}_i|\mathbf{y}) + \log P(\mathbf{y}) \\ = \sum_{i=1}^{n} \left[-\frac{1}{2} (\boldsymbol{\phi}_i - \mathbf{m} - \mathbf{U}\mathbf{y})^{\mathrm{T}} \boldsymbol{\Lambda}_{eq,i} \left(\boldsymbol{\phi}_i - \mathbf{m} - \mathbf{U}\mathbf{y}\right) \right] \\ + \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{y} + k, \end{aligned} \tag{7.10}$$

The posterior distribution of $\mathbf{y}$ given a set of i–vectors is again Gaussian:

$$\mathbf{y}|\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n \sim \mathcal{N}(\hat{\mathbf{y}}, \mathbf{P}^{-1}), \tag{7.11}$$

with parameters:

$$\mathbf{P} = \mathbf{I} + \sum_{i=1}^{n} \mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda}_{eq,i} \mathbf{U} \tag{7.12}$$

$$\hat{\mathbf{y}} = \mathbf{P}^{-1} \mathbf{U}^{\mathrm{T}} \sum_{i=1}^{n} \boldsymbol{\Lambda}_{eq,i} \left(\boldsymbol{\phi}_i - \mathbf{m}\right). \tag{7.13}$$

The likelihood of a set of segments belonging to the same speaker can be written as

$$P(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n|H_s) = \frac{P(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n|\mathbf{y}_0) P(\mathbf{y}_0)}{P(\mathbf{y}_0|\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n)}, \tag{7.14}$$

which is the same form as in the original PLDA and setting $\mathbf{y}_0 = \mathbf{0}$ for convenience will produce a result similar to equation (6.19). Using (7.11), and (7.5) we finally get

$$\begin{aligned} &\log P(\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_n|H_s) = \\ &\sum_{i=1}^{n} \left[\frac{1}{2} \log |\boldsymbol{\Lambda}_{eq,i}| - \frac{M}{2} \log 2\pi - \frac{1}{2} (\boldsymbol{\phi}_i - \mathbf{m})^{\mathrm{T}} \boldsymbol{\Lambda}_{eq,i} (\boldsymbol{\phi}_i - \mathbf{m}) \right] \\ &- \frac{1}{2} \log |\mathbf{P}| + \frac{1}{2} \hat{\mathbf{y}}^{\mathrm{T}} \mathbf{P} \hat{\mathbf{y}} - \frac{S}{2} \log 2\pi, \end{aligned} \tag{7.15}$$

where M is the i–vector dimension, and S is the speaker factor dimension. Note again, the difference to the standard PLDA lies in the segment-based $\boldsymbol{\Lambda}_{eq,i}$, which greatly affects the computational complexity of scoring. We will compare the complexity to the classical PLDA in section 7.5.

7.4 Parameter Estimation

The model presented in (7.7) allows obtaining a simple expression for computing the log–likelihood ratio of a speaker recognition trial. However, it does not allow the update formulas to be easily derived. An equivalent expression of (7.7), where the contributions of the i–vector posterior covariance and of the residual noise are decoupled, is more suitable for the estimation of model parameters [Kenny et al., 2013]. To this extent, the segment–dependent residual term $\overline{\mathbf{E}}_i$ can be written as:

$$\overline{\mathbf{E}}_i = \mathbf{C}_i \mathbf{X}_i + \mathbf{E}, \tag{7.16}$$

where $\mathbf{C}_i$ is is given by the Cholesky decomposition $\mathbf{C}_i \mathbf{C}_i^{\mathrm{T}} = \boldsymbol{\Gamma}_i^{-1}$, $\mathbf{X}_i$ is a standard Gaussian distributed random variable, $\mathbf{X}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\mathbf{E}$ is the PLDA residual term introduced in (6.12). The corresponding PLDA model is then given by:

$$\boldsymbol{\phi}_i = \mathbf{m} + \mathbf{U}\mathbf{y} + \mathbf{C}_i \mathbf{x}_i + \mathbf{e}_i, \tag{7.17}$$

where $\mathbf{x}_i$ is a realization of $\mathbf{X}_i$. It is worth noting that (7.17) formally corresponds to the PLDA model in (6.1) with the channel subspace matrix $\mathbf{V}$ replaced by a segment–dependent matrix $\mathbf{C}_i$. The same steps to derive the EM algorithm for the PLDA model (6.1) can be easily modified to estimate the parameters of the FPD–PLDA model. The details of the derivation of the EM algorithm can be found in [Kenny et al., 2013] or [Brümmer, 2010a] with modifications related to this model. We will again follow [Brümmer, 2010a] to derive the re-estimation formulas.

Let us start with observing the speaker s as an i–vector $\boldsymbol{\phi}_{s,i}$ and decomposing it according to (7.17). For the simplicity, we will assume zero mean $\mathbf{m}$ as i–vectors can be centered:

$$\boldsymbol{\phi}_{s,i} = \mathbf{U}\mathbf{y}_s + \mathbf{C}_{s,i} \mathbf{x}_{s,i} + \mathbf{e}_{s,i}. \tag{7.18}$$

The parameters of the model are $\boldsymbol{\lambda} = \langle \mathbf{U}, \boldsymbol{\Lambda} \rangle$, where $\boldsymbol{\Lambda}$ is the precision matrix of the posterior distribution of $\mathbf{e}$ defined in (6.12). Matrices $\mathbf{C}_{s,i}$ are given for each segment and represent the uncertainty in i–vector estimate.

7.4.1 Data

We will use similar notation as in Section 6.2.2. $\mathcal{Y}$ is the set of all speakers and for each speaker $s \in \mathcal{Y}$, we have $i = 1 \ldots n_s$ observations. Each observation is represented by a single i–vector $\boldsymbol{\phi}_{s,i}$ and a segment-depend matrix $\mathbf{C}_{s,i}$. Let us stack all i–vectors of speaker s into a $M \times n_s$ matrix

$$\boldsymbol{\Phi}_s = \left[\boldsymbol{\phi}_1 \ldots \boldsymbol{\phi}_{s,n_s} \right], \tag{7.19}$$

where M is the dimensionality of the i–vector and also let the $\mathbf{X}_s$ be the matrix of all hidden variables $\mathbf{x}_{s,i}$

$$\mathbf{X}_s = \left[\mathbf{x}_{s,1} \ldots \mathbf{x}_{s,n_s} \right]. \tag{7.20}$$

We will also need to define sufficient statistics as in the case with standard PLDA, but here, assuming zero mean, the formulas will simplify. The global zero-order statistics for all observations and speakers are given as

$$N = \sum_{s \in \mathcal{Y}} n_s, \tag{7.21}$$

and the second-order statistics for all observations are given as

$$\mathbf{S} = \sum_{s \in \mathcal{Y}} \sum_{i=1}^{n_s} \boldsymbol{\phi}_{s,i} \boldsymbol{\phi}_{s,i}^{\mathrm{T}}. \tag{7.22}$$

7.4.2 Log-Likelihood

For speaker s, the log-likelihood of the data is given by

$$\begin{aligned} \log p(\boldsymbol{\Phi}_s | \mathbf{y}_s, \mathbf{X}_s, \boldsymbol{\lambda}) &= \sum_{i=1}^{n_s} \log \mathcal{N}(\phi_{s,i} | \mathbf{U}\mathbf{y}_s + \mathbf{C}_{s,i}\mathbf{x}_{s,i}, \boldsymbol{\Lambda}^{-1}) \qquad (7.23) \\ &= \sum_{i=1}^{n_s} \Bigg(-\frac{1}{2} \boldsymbol{\phi}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \boldsymbol{\phi}_{s,i} + \boldsymbol{\phi}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U} \mathbf{y}_s + \boldsymbol{\phi}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{C}_{s,i} \mathbf{x}_{s,i} \\ &\quad -\frac{1}{2} \mathbf{y}_s^{\mathrm{T}} \mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U} \mathbf{y}_s - \mathbf{y}_s^{\mathrm{T}} \mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{C}_{s,i} \mathbf{x}_{s,i} - \frac{1}{2} \mathbf{x}_{s,i}^{\mathrm{T}} \mathbf{C}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{C}_{s,i} \mathbf{x}_{s,i} \\ &\quad -\frac{M}{2} \log(2\pi) - \frac{1}{2} \log |\boldsymbol{\Lambda}| \Bigg). \end{aligned}$$

7.4.3 Hidden Variable Distributions

The joint prior for the hidden variables and speaker s is defined as

$$\begin{aligned} p(\mathbf{X}_s, \mathbf{y}_s) &= p(\mathbf{X}_s)\, p(\mathbf{y}_s) \\ \log p(\mathbf{X}_s, \mathbf{y}_s) &= -\frac{1}{2} \mathbf{y}_s^{\mathrm{T}} \mathbf{y}_s - \frac{1}{2} \mathrm{tr}(\mathbf{X}_s^{\mathrm{T}} \mathbf{X}_s) + \mathrm{const.} \end{aligned} \tag{7.24}$$

To define the posteriors of the hidden variables, we will define auxiliary substitutions. Note that here, in contrast with standard PLDA, the terms will be segment-dependent, as we are given segment-dependent matrix $\mathbf{C}_i$ in our model:

$$\mathbf{J}_{s,i} = \mathbf{C}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U} \tag{7.25}$$

$$\mathbf{K}_{s,i} = \mathbf{C}_{s,i}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{C}_{s,i} + \mathbf{I}. \tag{7.26}$$

The joint posterior probability of the hidden variables can be assembled from two factors as

$$p(\mathbf{X}_s, \mathbf{y}_s | \boldsymbol{\Phi}_s, \boldsymbol{\lambda}) = p(\mathbf{X}_s | \mathbf{y}_s, \boldsymbol{\Phi}_s, \boldsymbol{\lambda})\, p(\mathbf{y}_s | \Phi_s, \boldsymbol{\lambda}). \tag{7.27}$$

The posterior probability $p(\mathbf{y}_s | \boldsymbol{\Phi}_s, \boldsymbol{\lambda}) = \mathcal{N}(\mathbf{y}_s | \hat{\mathbf{y}}_s, \mathbf{P}_s^{-1})$ was already defined in (7.11). The corresponding precision matrix $\mathbf{P}_s$ and mean $\hat{\mathbf{y}}_s$ are defined by equations (7.12) and (7.13).

To define the posterior probability $\log p(\mathbf{x}_{s,i} | \mathbf{y}_s, \boldsymbol{\Phi}_s, \boldsymbol{\lambda})$, we take the joint probability $\log p(\boldsymbol{\Phi}_s, \mathbf{y}_s, \mathbf{x}_s)$. By summing (7.23) and (7.24) and ignoring the terms not dependent on $\mathbf{x}_{s,i}$, we obtain

$$\begin{aligned}\log p(\mathbf{x}_{s,i}|\mathbf{y}_s, \mathbf{\Phi}_s, \boldsymbol{\lambda}) \propto\ & \boldsymbol{\phi}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{C}_{s,i}\mathbf{x}_{s,i} - \mathbf{y}_s^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{C}_{s,i}\mathbf{x}_{s,i} \\ & - \frac{1}{2}\mathbf{x}_{s,i}^{\mathrm{T}}\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{C}_{s,i}\mathbf{x}_{s,i} - \frac{1}{2}\mathbf{x}_{s,i}^{\mathrm{T}}\mathbf{x}_{s,i} \\ =\ & \mathbf{x}_{s,i}^{\mathrm{T}}(\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\boldsymbol{\phi}_{s,i} - \mathbf{J}\mathbf{y}_s) - \frac{1}{2}\mathbf{x}_{s,i}^{\mathrm{T}}\mathbf{K}_{s,i}\mathbf{x}_{s,i}. \qquad (7.28)\end{aligned}$$

By using completion of squares and by summing over all segments $\mathbf{x}_{s,i}$, we can write

$$\log p(\mathbf{X}_s|\mathbf{y}_s, \mathbf{\Phi}_s, \boldsymbol{\lambda}) = \sum_{i=1}^{n_s} \mathcal{N}(\mathbf{x}_{s,i}|\hat{\mathbf{x}}_{s,i}, \mathbf{K}_{s,i}^{-1}), \qquad (7.29)$$

where

$$\hat{\mathbf{x}}_{s,i} = \mathbf{K}_{s,i}^{-1}(\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\boldsymbol{\phi}_{s,i} - \mathbf{J}_{s,i}\mathbf{y}_s). \qquad (7.30)$$

7.4.4 Evidence

The log evidence, which is guaranteed to increase in every consecutive iteration of the EM algorithm, is useful to monitor the convergence of the training. It is defined as a marginal log-likelihood of the observations given the system parameters:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\lambda}) &= \sum_{s\in\mathcal{Y}} \log p(\mathbf{\Phi}_s|\boldsymbol{\lambda}) \\ &= \sum_{s\in\mathcal{Y}} \log \frac{p(\mathbf{\Phi}_s|\mathbf{y}_s, \mathbf{X}_s, \boldsymbol{\lambda})\, p(\mathbf{y}_s)\, p(\mathbf{X}_s)}{p(\mathbf{X}_s|\mathbf{y}_s, \mathbf{\Phi}_s, \boldsymbol{\lambda})\, p(\mathbf{y}_s|\mathbf{\Phi}_s, \boldsymbol{\lambda})}. \qquad (7.31)\end{aligned}$$

7.4.5 E-step

The EM auxiliary function is

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\lambda}, \boldsymbol{\lambda}_0) &= \sum_{s\in\mathcal{Y}} \left\langle \log p(\mathbf{\Phi}_s|\mathbf{y}_s, \mathbf{X}_s, \boldsymbol{\lambda})\right\rangle_{\mathbf{X}_s,\mathbf{y}_s|\mathbf{\Phi}_s,\boldsymbol{\lambda}_0} \qquad (7.32) \\ &= \frac{N}{2}\log|\mathbf{\Lambda}| - \frac{1}{2}\mathrm{tr}(\mathbf{S}\mathbf{\Lambda}) \\ &- \frac{1}{2}\sum_{s\in\mathcal{Y}}\sum_{i=1}^{n_s} \mathrm{tr}(\mathbf{R}_{xx_{s,i}}\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{C}_{s,i} + \mathbf{R}_{xy_{s,i}}\mathbf{U}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{C}_{s,i} + \mathbf{R}_{xy_{s,i}}^{\mathrm{T}}\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{U} + \mathbf{R}_{yy_s}\mathbf{U}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{U}) \\ &+ \frac{1}{2}\sum_{s\in\mathcal{Y}}\sum_{i=1}^{n_s} \mathrm{tr}(\mathbf{T}_{\mathbf{y}_{s,i}}\mathbf{\Lambda}\mathbf{U} + \mathbf{U}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{T}_{\mathbf{y}_{s,i}}^{\mathrm{T}} + \mathbf{T}_{\mathbf{x}_{s,i}}\mathbf{\Lambda}\mathbf{C}_{s,i} + \mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{T}_{\mathbf{x}_{s,i}}^{\mathrm{T}}),\end{aligned}$$

where auxiliary substitutions are:

$$\begin{aligned}\mathbf{T}_{\mathbf{y}_{s,i}} &= \hat{\mathbf{y}}_s\boldsymbol{\phi}_{s,i}^{\mathrm{T}} \qquad (7.33) \\ \mathbf{T}_{\mathbf{x}_{s,i}} &= \left\langle \hat{\mathbf{x}}_{s,i}(\mathbf{y}_{s,i})^{\mathrm{T}}\right\rangle_{\mathbf{y}_s|\mathbf{\Phi}_s,\boldsymbol{\lambda}_0} \boldsymbol{\phi}_{s,i}^{\mathrm{T}} \qquad (7.34) \\ &= \mathbf{K}_{s,i}^{-1}(\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\boldsymbol{\phi}_{s,i}\boldsymbol{\phi}_{s,i}^{\mathrm{T}} - \mathbf{J}_{s,i}\mathbf{T}_{\mathbf{y}_{s,i}}).\end{aligned}$$

To define the cross-correlations $\mathbf{R}_{yy_s}$, $\mathbf{R}_{xy_{s,i}}$ and $\mathbf{R}_{xx_{s,i}}$ it is useful to realize that the joint distribution over two variables can be expressed (see section 2.3.3 of [Bishop, 2006]) as a Gaussian

$$p(\mathbf{x}_{s,i}, \mathbf{y}_s | \mathbf{\Phi}, \boldsymbol{\lambda}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_{s,i} \\ \mathbf{y}_s \end{bmatrix} \middle| \begin{bmatrix} \overline{\mathbf{x}}_{s,i} \\ \hat{\mathbf{y}}_s \end{bmatrix}, \begin{bmatrix} \mathbf{\Sigma}_{xx_{s,i}} & \mathbf{\Sigma}_{xy_{s,i}} \\ \mathbf{\Sigma}^{\mathrm{T}}_{xy_{s,i}} & \mathbf{P}_s^{-1} \end{bmatrix} \right), \tag{7.35}$$

where

$$\overline{\mathbf{x}}_{s,i} = \mathbf{K}_{s,i}^{-1}(\mathbf{C}_{s,i}\mathbf{\Lambda}\boldsymbol{\phi}_{s,i} - \mathbf{J}_{s,i}\hat{\mathbf{y}}_s) \tag{7.36}$$

$$\mathbf{\Sigma}_{xx_{s,i}} = \mathbf{K}_{s,i}^{-1} + \mathbf{K}_{s,i}^{-1}\mathbf{J}_{s,i}\mathbf{P}_s^{-1}\mathbf{J}_{s,i}\mathbf{K}_{s,i}^{-1} \tag{7.37}$$

$$\mathbf{\Sigma}_{xy_{s,i}} = -\mathbf{K}_{s,i}^{-1}\mathbf{J}_{s,i}\mathbf{P}_s^{-1} \tag{7.38}$$

and finally we get

$$\mathbf{R}_{yy_s} = \left\langle \mathbf{y}_s\mathbf{y}_s^{\mathrm{T}} \right\rangle = \mathbf{P}_s^{-1} + \hat{\mathbf{y}}_s\hat{\mathbf{y}}_s^{\mathrm{T}} \tag{7.39}$$

$$\begin{aligned} \mathbf{R}_{xy_{s,i}} &= \left\langle \mathbf{x}_{s,i}\mathbf{y}_{s,i}^{\mathrm{T}} \right\rangle \\ &= \mathbf{\Sigma}_{xy_{s,i}} + \overline{\mathbf{x}}_{s,i}\overline{\mathbf{x}}_{s,i}^{\mathrm{T}} \\ &= \mathbf{K}_{s,i}^{-1}(\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{T}_{\mathbf{y}_{s,i}} - \mathbf{J}_{s,i}\mathbf{R}_{yy_s}) \end{aligned} \tag{7.40}$$

$$\begin{aligned} \mathbf{R}_{xx_{s,i}} &= \left\langle \mathbf{x}_{s,i}\mathbf{x}_{s,i}^{\mathrm{T}} \right\rangle \\ &= \mathbf{\Sigma}_{xx_{s,i}} + \overline{\mathbf{x}}_{s,i}\overline{\mathbf{x}}_{s,i}^{\mathrm{T}} \\ &= \mathbf{K}_{s,i}^{-1}(\mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\boldsymbol{\phi}_{s,i}\boldsymbol{\phi}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{C}_{s,i} - \mathbf{C}_{s,i}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{T}_{\mathbf{y}_{s,i}}^{\mathrm{T}}\mathbf{J}_{s,i}^{\mathrm{T}} - \\ &\quad - \mathbf{J}_{s,i}\mathbf{T}_{\mathbf{y}_{s,i}}\mathbf{\Lambda}\mathbf{C}_{s,i} + \mathbf{J}_{s,i}\mathbf{R}_{yy_s}\mathbf{J}_{s,i}^{\mathrm{T}})\mathbf{K}_{s,i}^{-1} + \mathbf{K}_{s,i}^{-1}. \end{aligned} \tag{7.41}$$

7.4.6 M-step

Taking the derivative of EM auxiliary function (7.32) with respect to $\mathbf{U}$ and setting it to zero yields the maximum-likelihood re-estimation of $\mathbf{U}$ as

$$\mathbf{U} = \sum_{s\in\mathcal{Y}}\sum_{i=1}^{n_s}\left(\mathbf{T}_{\mathbf{y}_{s,i}}^{\mathrm{T}} - \mathbf{C}_{s,i}\mathbf{R}_{xy_{s,i}}\right)\left(\sum_{s\in\mathcal{Y}} n_s\mathbf{R}_{yy_s}\right)^{-1}. \tag{7.42}$$

Differentiating (7.32) w.r.t. $\mathbf{\Lambda}$ gives

$$\begin{aligned} \mathbf{\Lambda}^{-1} = \frac{\mathbf{S}}{N} + \frac{1}{N}\sum_{s\in\mathcal{Y}}\sum_{i=1}^{n_s}\Big(&\mathbf{C}_{s,i}\mathbf{R}_{xx_{s,i}}\mathbf{C}_{s,i}^{\mathrm{T}} + \mathbf{C}_{s,i}\mathbf{R}_{xy_{s,i}}\mathbf{U}^{\mathrm{T}} + \mathbf{U}\mathbf{R}_{xy_{s,i}}^{\mathrm{T}}\mathbf{C}_{s,i}^{\mathrm{T}} \\ &+\mathbf{U}\mathbf{R}_{yy_s}\mathbf{U}^{\mathrm{T}} - \mathbf{U}\mathbf{T}_{y_{s,i}} - \mathbf{T}_{y_{s,i}}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} - \mathbf{C}_{s,i}\mathbf{T}_{x_{s,i}} - \mathbf{T}_{x_{s,i}}^{\mathrm{T}}\mathbf{C}_{s,i}^{\mathrm{T}}\Big). \end{aligned} \tag{7.43}$$

Finally, we can also apply minimum divergence step in the same way as with the classical PLDA. We will transform the maximum likelihood estimate of the $\mathbf{U}$ in such a way that we force the speaker factor posteriors to be distributed according to the standard normal prior. We accumulate

$$\mathbf{A} = \frac{1}{|\mathcal{Y}|}\sum_{s\in\mathcal{Y}}\sum_{i=1}^{n_s}\left\langle \mathbf{y}_{s,i}\mathbf{y}_{s,i}^{\mathrm{T}} \right\rangle = \frac{1}{|\mathcal{Y}|}\sum_{s\in\mathcal{Y}}\mathbf{P}_s^{-1} + \hat{\mathbf{y}}_s\hat{\mathbf{y}}_s^{\mathrm{T}} \tag{7.44}$$

and the minimum-divergence re-estimation of $\mathbf{U}$ is given by

$$\mathbf{U} \leftarrow \mathbf{U}\,\mathrm{chol}(\mathbf{A})^{\mathrm{T}}. \tag{7.45}$$

7.5 Complexity Analysis

The straightforward implementations of classical PLDA and FPD–PLDA have similar computational complexity. However, in practical scenarios, some of the terms required for the evaluation of the PLDA log–likelihood ratio (6.10) can be pre–computed. These pre-computations allow for fast scoring, at the cost of a slight increase of the memory requirements for the PLDA model and for the target models. Unfortunately, some of these optimizations cannot be done for FPD–PLDA, which is thus a more accurate but slower approach. In the following we analyze the computational complexity of PLDA and FPD–PLDA implementations optimized for the most common scenario. This scenario is the speaker detection task where the system has to score several test sets, whose numbers of segments are known in advance, against a fixed set of target speakers. In particular, each set of segments of a single test speaker has to be verified against the segments of a known, fixed set of target speakers. Since all targets are known in advance, target–dependent (enrollment-depend) optimizations can be performed offline. The NIST SRE 2012 evaluation [NIST, 2012] follows this protocol. However, even for the previous evaluations, where each trial had to be scored independently it was possible to speed–up the scoring for the complete evaluation, without violating its rules, because all enrollment segments were indeed known in advance.

In this scenario, as will be shown in sub-sections 7.5.2 and 7.5.3, a smart implementation of PLDA allows some of the terms required for the evaluation of the speaker verification log–likelihood ratio to be pre–computed, thus the per–trial scoring complexity is greatly reduced. Different optimizations are possible for FPD–PLDA depending on the duration of the enrollment segments. For short segments, FPD–PLDA does not allow the pre–computation of most of the terms of the scoring function, thus its complexity cannot be reduced. However, if the enrollment segments are long enough, their i–vector posteriors can be safely approximated by their MAP point estimates, and the per–trial complexity of the proposed technique can be reduced.

7.5.1 Log–likelihood Computation

The complexity of the log–likelihood computation accounts for two separate contributions. The first contribution is the complexity of operations that can be independently performed on target or test sets, which will be referred to as per–enroll and per–test terms, respectively. The second contribution is the per–trial complexity, i.e. the complexity of the terms which jointly involve the enrollment and the test sets. This distinction is not relevant for the naïve scoring implementations. Is relevant, instead, in the "fixed set of target speakers scenarios" because the per–enroll terms can be pre–computed, and per–test terms need to be computed only once regardless of the number of target speakers.

We will analyze both per–test and per–trial complexity of the PLDA and FPD–PLDA models. It is worth noting that the complexity of a complete system should account also for the complexity of the extraction of the acoustic features and of the i–vectors. The computation of the i–vector covariance matrix, for each segment, has complexity $O(M^3)$ [Glembek et al., 2011], which, as we will see, dominates the other costs.

Since we compute the speaker variable $\mathbf{y}$ posteriors on different sets of segments, we explicitly condition the parameters of the posterior distributions of $\mathbf{y}$ (7.11) to a generic set G as:

$$\begin{aligned}\mathbf{P}_G &= \mathbf{I} + \sum_{i\in G} \mathbf{U}^{\mathrm{T}}\boldsymbol{\Lambda}_{eq,i}\mathbf{U} \\ \hat{\mathbf{y}}_G &= \mathbf{P}_G^{-1}\mathbf{U}^{\mathrm{T}} \sum_{i\in G} \boldsymbol{\Lambda}_{eq,i}\left(\boldsymbol{\phi}_i - \mathbf{m}\right). \end{aligned} \tag{7.46}$$

The index of the sum in this equation, and in the following equations, is to be interpreted as running over all the segments of the set. Replacing (7.15) in (6.10), the speaker verification log–likelihood ratio for an enrollment set E and a test set T can be written as:

$$\begin{aligned} llr(E,T) &= \log \frac{l(E,T|H_s)}{l(E|H_s)l(T|H_s)} \\ &= -\frac{1}{2}\log\left|\mathbf{P}_{(E,T)}\right| + \frac{1}{2}\hat{\mathbf{y}}_{(E,T)}^{\mathrm{T}}\mathbf{P}_{(E,T)}\hat{\mathbf{y}}_{(E,T)} \\ &\quad + \frac{1}{2}\log\left|\mathbf{P}_{(E)}\right| - \frac{1}{2}\hat{\mathbf{y}}_{(E)}^{\mathrm{T}}\mathbf{P}_{(E)}\hat{\mathbf{y}}_{(E)} \\ &\quad + \frac{1}{2}\log\left|\mathbf{P}_{(T)}\right| - \frac{1}{2}\hat{\mathbf{y}}_{(T)}^{\mathrm{T}}\mathbf{P}_{(T)}\hat{\mathbf{y}}_{(T)} \\ &\quad + \frac{S}{2}\log 2\pi \\ &= \sigma(E,T) - \sigma(E) - \sigma(T) + \frac{S}{2}\log 2\pi, \end{aligned} \tag{7.47}$$

where the scoring function σ is defined as:

$$\sigma(G) = -\frac{1}{2}\log\left|\mathbf{P}_{(G)}\right| + \frac{1}{2}\hat{\mathbf{y}}_{(G)}^{\mathrm{T}}\mathbf{P}_{(G)}\hat{\mathbf{y}}_{(G)}. \tag{7.48}$$

Since the computation of $\sigma(E)$ and $\sigma(T)$ cannot be more expensive than the computation of $\sigma(E,T)$, we restrict our analysis to this term of the log–likelihood ratio.

7.5.2 Complexity of the Standard PLDA

As shown in Section 7.3, standard PLDA corresponds to a FPD–PLDA with $\boldsymbol{\Gamma}_i^{-1} = \mathbf{0}$ for all i–vectors. Thus, $\boldsymbol{\Lambda}_{eq,i} = \boldsymbol{\Lambda}$ for all i–vectors, and the speaker variable posterior parameters become:

$$\begin{aligned}\mathbf{P}_{(E,T)} &= \mathbf{I} + (n_E + n_T)\mathbf{U}^{\mathrm{T}}\boldsymbol{\Lambda}\mathbf{U} \\ \hat{\mathbf{y}}_{(E,T)} &= \mathbf{P}_{(E,T)}^{-1}\mathbf{U}^{\mathrm{T}}\boldsymbol{\Lambda}\left(\sum_{i\in E}\left(\boldsymbol{\phi}_i - \mathbf{m}\right) + \sum_{i\in T}\left(\boldsymbol{\phi}_i - \mathbf{m}\right)\right)\end{aligned}$$

$$= \mathbf{P}_{(E,T)}^{-1} \left(\mathbf{F}_E + \mathbf{F}_T \right) , \tag{7.49}$$

where n_E and n_T are the numbers of enrollment and test segments respectively, $\mathbf{F}_E$ and $\mathbf{F}_T$ are the projected first order statistics

$$\mathbf{F}_E = \mathbf{M} \sum_{i \in E} \left(\boldsymbol{\phi}_i - \mathbf{m} \right) , \;\; \mathbf{F}_T = \mathbf{M} \sum_{i \in T} \left(\boldsymbol{\phi}_i - \mathbf{m} \right) , \tag{7.50}$$

and $\mathbf{M} = \mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda}$ is an $S \times M$ matrix with S and M being the size of the speaker subspace and dimensionality of i–vectors, respectively. Using these definitions, the scoring function $\sigma(E,T)$ can be rewritten as:

$$\begin{aligned} \sigma(E,T) = & -\frac{1}{2} \log \left| \mathbf{P}_{(E,T)} \right| + \mathbf{F}_E^{\mathrm{T}} \mathbf{P}_{(E,T)}^{-1} \mathbf{F}_T \\ & + \frac{1}{2} \mathbf{F}_T^{\mathrm{T}} \mathbf{P}_{(E,T)}^{-1} \mathbf{F}_T + \frac{1}{2} \mathbf{F}_E^{\mathrm{T}} \mathbf{P}_{(E,T)}^{-1} \mathbf{F}_E . \end{aligned} \tag{7.51}$$

Computing the projected statistics (7.50) has complexity $O(NM + MS)$, where N is the number of speech segments in the set. It is worth noting that the $\mathbf{F}_E$ and $\mathbf{F}_T$ statistics are per-enroll and per–test computations because they can be computed for the enrollment and test sets independently.

Naïve Scoring Implementation

The computation of the score function $\sigma(E,T)$, given the $\mathbf{F}_G$ statistics, requires computing $\mathbf{P}_{(E,T)}^{-1}$ and its log–determinant. These computations have complexity $O(S^3)$ because, for standard PLDA, the term $\mathbf{U}^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{U}$ can be pre-computed. Given $\mathbf{P}_{(E,T)}^{-1}$, scoring $\sigma(E,T)$ has complexity $O(S^2)$. The same considerations apply to the less expensive computation of $\sigma(E)$ and $\sigma(T)$. Thus, the overall per–trial complexity is $O(S^3)$.

Speaker Detection with Known Enrollment Sets

In the naïve implementation, the computation and inversion of $\mathbf{P}_{(E,T)}$ dominates the scoring costs. However, in standard PLDA this factor varies only with the number $(n_T + n_E)$ of the enrollment and test segments (7.49). When each set of enrollment segments E_k, and the number of test segments n_T, are known, it is possible to pre–compute the corresponding $\mathbf{P}_{(E_k,T)}^{-1}$, and its log–determinant. Moreover, since the statistics $\mathbf{F}_{E_k}$ are also known in advance, the terms of the scoring function $\frac{1}{2} \mathbf{F}_{E_k}^{\mathrm{T}} \mathbf{P}_{(E_k,T)}^{-1}$ can be pre–computed. It is worth noting that these terms are small S–sized vectors. Since the term depending only on the test statistics $\mathbf{F}_T$ must be evaluated just once for the whole set of K targets, its computation has a per–test, rather than a per–trial, cost. Every function $\sigma(E_k,T)$ can be computed with complexity $O(S)$, each term $\sigma(E_k)$ can be easily pre–computed. Given the statistics, the term $\sigma(T)$ has a per–enroll and per–test complexity of $O(S^2)$. The overall per-enroll and per–test cost, including statistics computations, is then $O(NM + MS)$, whereas the per–trial cost is $O(S)$.

7.5.3 Full Posterior Distribution PLDA

The main difference between the standard PLDA and the FPD–PLDA approach is that in PLDA $\mathbf{P}_{(E,T)}$ depends just on the number of i–vectors in the set (7.49), whereas in FPD–PLDA it also depends on the covariance of each i–vector in the test set T (see (7.46)). This does not allow for applying the same optimizations as illustrated in the previous section to the FPD–PLDA.

The speaker variable posterior parameters can still be written as:

$$\begin{aligned} \mathbf{P}_{(E,T)} &= \mathbf{I} + (\boldsymbol{\Lambda}_{eq,E} + \boldsymbol{\Lambda}_{eq,T}) \\ \hat{\mathbf{y}}_{(E,T)} &= \mathbf{P}^{-1}\left(\mathbf{F}_{eq,E} + \mathbf{F}_{eq,T}\right), \end{aligned} \tag{7.52}$$

where

$$\begin{aligned} \mathbf{F}_{eq,G} &= \mathbf{U}^{\mathrm{T}} \sum_{i \in G} \boldsymbol{\Lambda}_{eq,i} \left(\boldsymbol{\phi}_i - \mathbf{m}\right) \\ \boldsymbol{\Lambda}_{eq,G} &= \mathbf{U}^{\mathrm{T}} \left(\sum_{i \in G} \boldsymbol{\Lambda}_{eq,i} \right) \mathbf{U}, \end{aligned} \tag{7.53}$$

and the scoring function $\sigma(E,T)$ can be rewritten as:

$$\begin{aligned} \sigma(E,T) = &- \frac{1}{2} \log \left| \mathbf{P}^{-1}_{(E,T)} \right| + \frac{1}{2} \mathbf{F}^{\mathrm{T}}_{eq,E} \mathbf{P}^{-1}_{(E,T)} \mathbf{F}_{eq,E} \\ &+ \frac{1}{2} \mathbf{F}^{\mathrm{T}}_{eq,T} \mathbf{P}^{-1}_{(E,T)} \mathbf{F}_{eq,T} + \mathbf{F}^{\mathrm{T}}_{eq,E} \mathbf{P}^{-1}_{(E,T)} \mathbf{F}_{eq,T}. \end{aligned} \tag{7.54}$$

Computing the posterior parameters (7.52) has a complexity $O(NM^3 + M^2S)$, mainly due to the computation of $\boldsymbol{\Lambda}_{eq,i}$, and is much higher than the $O(NM + MS)$ complexity of standard PLDA approach. However, these computations are required only for a new target or test speaker. These costs are comparable to the costs $O(NM^3)$ of the i–vector extraction [Glembek et al., 2011]. Given the statistics, $\mathbf{P}_{(E,T)}$ can be computed with complexity $O(S^2)$ and its inversion complexity is $O(S^3)$. The computation of the remaining terms requires $O(S^2)$, thus the overall per–trial complexity is $O(S^3)$. Since the posterior parameter $\mathbf{P}_{(E,T)}$ cannot be pre–computed as in standard PLDA, the per–trial complexity is the same also for the fixed set of target speakers scenarios.

7.5.4 Asymmetric Full Posterior Distribution PLDA

In some applications, the target speaker segments have long enough duration, so that replacing the corresponding i–vector posterior distribution by a MAP point estimate has a negligible impact on the term $\boldsymbol{\Lambda}_{eq,E}$. In this case, it is possible to narrow the complexity gap between standard PLDA and FPD–PLDA, because the i–vector covariance is taken into account only for the test segments. Thus, we refer to this approach as Asymmetric Full Posterior Distribution PLDA. Since MAP–approximated i–vectors are used for the target speakers, the computational complexity of $\sigma(E)$ becomes equivalent to the one of the standard PLDA. The per–trial complexity with respect to the standard FPD–PLDA approach can be reduced because the same test set is scored against a fixed set of target

Table 7.1: Comparison of the log–likelihood computation complexity for three implementations of PLDA. Per–segment costs should be multiplied by the number of segments N of a given speaker. Per–speaker costs do not depend on the number of speaker segments. These costs refer to PLDA only, without considering the contribution of i–vector extraction.

System	Per–segment costs	Per–enroll, per-test fixed costs	Per–trial costs
Naïve PLDA	M	MS	S^3
Optimized PLDA	M	MS	S
Standard FPD–PLDA	M^3	M^2S	S^3
Asymmetric FPD–PLDA	M^3	M^2S	S^2

speakers. In particular, the covariance of the posterior of the speaker identity variable

$$\mathbf{P}_{(E,T)} = \mathbf{I} + n_E \mathbf{U}^{\mathrm{T}} \mathbf{\Lambda} \mathbf{U} + \sum_{i \in T} \mathbf{U}^{\mathrm{T}} \mathbf{\Lambda}_{eq,i} \mathbf{U} \tag{7.55}$$

depends only on the test i–vector covariance, and on the number of enrollment segments. If the number of target segments per speaker is fixed, computing the term $\mathbf{P}^{-1}_{(E_k,T)}$ for each target speaker becomes a per–test cost because it can be computed only once. Computing the scoring function, given $\mathbf{P}^{-1}_{(E_k,T)}$, has thus complexity $O(S^2)$.

Table 7.1 summarizes the results presented in this section. The costs have been divided into per–segment costs, depending on the number N of segments in the set, per–enroll and per-test fixed costs, and the per–trial costs.

The FPD-PLDA approach has a notably higher complexity that standard PLDA. The Asymmetric FPD-PLDA reduces the per–trial cost by a factor S, speeding–up the scoring computation when the number of target speakers is high. However, the duration of the enrollment segments affects the accuracy of the approximation, and possibly the performance gain with respect to the standard PLDA.

Chapter 8

I–vector Pre-Processing

The need of normalizing or transforming data, which are taken as inputs to various models, always originates from the fact that the data do not comply to the model's assumptions. It is not different in the case of i–vectors and various probabilistic models (in particular different variants of PLDA) for speaker recognition.

We assume that i–vectors are standard-normal distributed and both speaker and channel effects modeled by the Gaussian PLDA are additive, statistically independent and normally distributed. In [Kenny, 2010], Patrick Kenny clearly demonstrated that these assumptions are not satisfied, which leads to a sub-optimal performance of the model. Additionally, the score normalization was needed (S-norm, see Section 1.2.1) to obtain better results contradicting the intuition that a good generative model should produce well calibrated likelihood ratios which do not need to be further normalized.

The Gaussian assumptions effectively prohibited larger deviations from the mean — like the phenomena of having groups of outliers ("Black Swans") or observing channel effects very different from the ones present in the training data. Introducing the heavy-tailed version of the PLDA and relaxing the Gaussian assumptions by imposing Student's t priors on the hidden variables of the PLDA model brought significant improvements over the Gaussian PLDA model.

This model, although successful, is unfortunately burdened by its high complexity both in the training and scoring phase. It was obvious that in order to make the Gaussian PLDA competitive with HT-PLDA, the non-Gaussian behavior of the i–vectors needs to be eliminated or alternatively Gaussian behavior needs to be enforced.

A simple method of normalizing i–vectors to suit the Gaussian PLDA model was introduced in [Garcia-Romero, 2011]. The normalization generally consists of two steps: data whitening and length normalization. Whitening is the process where we enforce the total covariance matrix of i–vectors to be identity. The whitening can be performed as

$$\boldsymbol{\phi}_{\mathrm{wht}} = \mathbf{D}^{1/2}\mathbf{E}^{\mathrm{T}}\boldsymbol{\phi}, \tag{8.1}$$

where $\mathbf{E}$ and $\mathbf{D}$ are the orthogonal matrix of eigenvectors (in columns of $\mathbf{E}$) and diagonal matrix of eigenvalues of the total covariance matrix estimated on training i–vectors, respectively. Length normalization is a nonlinear transformation where we divide each i–vector by its norm and transform it to a vector of unit length:

$$\boldsymbol{\phi}_{\mathrm{norm}} = \frac{\boldsymbol{\phi}}{\|\boldsymbol{\phi}\|}. \tag{8.2}$$

8.1 Gaussianization of the Data

The superior results obtained with the HT-PLDA suggested that instead of assuming standard normal prior for the hidden variable in the i–vector model, we should consider that it follows Student's t-distribution. According to [Lyu et al., 2009], multivariate Student's t-distribution falls into the Elliptically Symmetric Densities (ESD) and therefore nonlinear transformation, which brings the samples of ESD family into a Gaussian distribution needs to be found. The technique proposed in [Lyu et al., 2009] called Radial Gaussianization (RG) consists of two steps. First, the ESD is transformed into a Spherically Symmetric Density (SSD) by a linear whitening transformation learned from the data samples of the ESD. Second, the distribution of the lengths of the whitened data $\boldsymbol{\phi}_{\mathrm{wht}}$ is non-linearly transformed. The idea behind the nonlinear length transformation $g(\|\boldsymbol{\phi}_{\mathrm{wht}}\|)$ is based on the fact that the lengths of vectors sampled from the multivariate Gaussian distribution follow a Chi distribution with D degrees of freedom, where D is the dimensionality of the vectors. The transformation is then given as a composition of the inverse cumulative Chi distribution with the cumulative distribution of the length random variable $r = \|\boldsymbol{\phi}_{\mathrm{wht}}\|$:

$$g(\|\boldsymbol{\phi}_{\mathrm{wht}}\|) = F_{\chi}^{-1} F_r(\|\boldsymbol{\phi}_{\mathrm{wht}}\|). \tag{8.3}$$

To accurately estimate the cumulative distribution of the length random variable, all of the data (especially evaluation data) need to be observed. This fact however violates the NIST SRE rules of processing each trial independently. For these reasons, the replacement of the second step in the gaussianization process by transforming the vector to unit length was proposed. It was shown in [Garcia-Romero, 2011] that performing the length normalization does not bring any performance degradation in comparison with properly estimating the nonlinear transformation from the data.

8.1.1 Length Normalization

Performing a transformation of the data into the unit length indeed again violates the Gaussian assumptions as the samples drawn from the high-dimensional standard normal Gaussians lie far away from the unit sphere. In fact, the samples are mostly present in a thin shell of a multidimensional sphere, of which distance from the origin is increasing with the dimensionality of data. If we are considering 600−dimensional i–vectors and knowing that the distribution of lengths of standard-normal distributed i–vectors follows Chi distribution, inner radius would be approximately 24 (see the mode of the Chi distribution in Figure 8.1).

When comparing the actual lengths of the i–vectors extracted from the training data and held out evaluation data, we observe completely different distributions of the lengths. In Figure 8.1, we present a situation of the i–vectors extracted for the Domain Adaptation

Challenge [MITLL, 2103]. There are three different datasets (training, adaptation and evaluation set) used in the Adaptation Challenge coming from various LDC data collections. The training set consists of all telephone calls from the all speakers taken from Switchboard-I and Switchboard-II (all phases) corpora. The adaptation set is composed of all telephone calls from all speakers taken from the NIST SRE data collections from years 2004, 2005, 2006 and 2008. Finally, the evaluation set is the telephone data from NIST SRE 2010 evaluations.

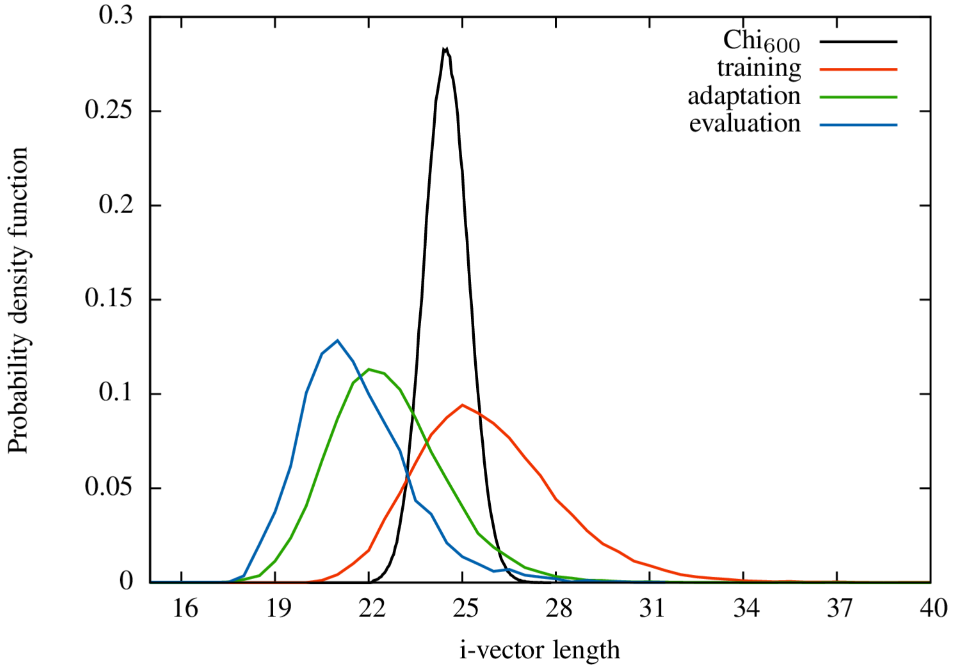


Figure 8.1: Histograms of the i–vector length distributions of three sets of Domain Adaptation Challenge. The probability density function of Chi distribution with 600 degrees of freedom depicted in black represents the distribution of 600 dimensional standard normal distributed vectors.

Not only we can observe a considerable shift in the lengths distributions of the individual databases, but all distributions have a longer right tail. The PDF of Chi distribution with 600 degrees of freedom representing the distribution of 600 dimensional standard normal distributed vectors is depicted in black color. As the i–vector extractor was trained on the training data, the i–vector length distribution of this dataset is closest to the expected distribution.

These shifts between datasets indeed lead to problems. As pointed out in [Garcia-Romero, 2011], the shift in the i–vector lengths would introduce a global scaling in the obtained scores (see equations 6.19 or 9.10). Scaling could be partly recovered by means of linear calibration. However, especially in the cases, when the evaluation data come from different sources, there would be more such scalings and one global calibration would not be sufficient to overcome this problem.

It is also interesting to observe the shift introduced by a different gender. Surprisingly, it is smaller than the shift between telephone databases. It is important to note, that the training data for i–vector extractor contained recordings from both genders. The situation is depicted on Figure 8.2, where the NIST SRE 2010 telephone data are split into female (f) and male (m) recordings.

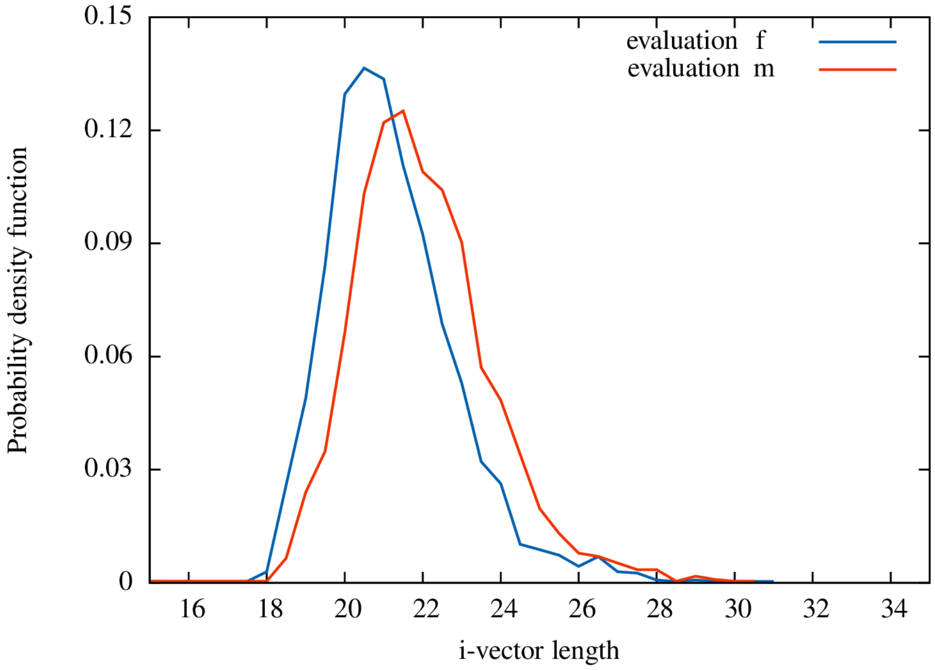


Figure 8.2: Histograms of the i–vector length distributions of female and male parts of the evaluation dataset from the Adaptation Challenge.

By performing normalization to unit length, we place all i–vectors on a surface of a common unit sphere and effectively greatly compress all distances between them. Also, we replace a distribution of their lengths by a constant. With a proper scaling, the constant could be even set into the mode of the Chi distribution, which in the end is not necessary. This way, we made the distribution of the i–vector lengths closer to the distribution of lengths of the i–vectors following standard-normal distribution. We also avoided problems with the score scaling. It is important to note, that before actual length normalization, we must ensure that the i–vectors are normalized to zero mean. Although zero mean of the i–vectors is also assumed by the i–vector extraction model, it is often not the case for i–vectors extracted from some held-out data. After all of these transformations, the PLDA is trained on normalized i–vectors. Alternatively, the cosine scoring can be directly performed.

8.2 Application to Full Posterior Distribution

This section presents the length normalization applied to the i–vector posterior distribution. A straightforward approach is to replace the i–vector distribution $\mathbf{W}|\mathcal{X}$ by $\widehat{\mathbf{W}} = \frac{\mathbf{W}|\mathcal{X}}{\|\mathbf{W}|\mathcal{X}\|}$, which forces all realizations of $\widehat{\mathbf{W}}$ to lie on the unit sphere. However, since the resulting random variable $\widehat{\mathbf{W}}$ would not be Gaussian distributed, it would not be possible to rely on the simple derivations of Section 6.2, and to avoid the higher complexity introduced by the use of a non Gaussian distribution.

Alternatively, the length normalization can be seen as a non–linear transformation $F(\boldsymbol{\phi}_0)$ of the observed i–vector $\boldsymbol{\phi}_0$, which can be approximated by its first order Taylor expansion around the i–vector itself. The expansion is given by:

$$F(\boldsymbol{\phi}) = F(\boldsymbol{\phi}_0) + J_F(\boldsymbol{\phi}_0)(\boldsymbol{\phi} - \boldsymbol{\phi}_0) + o(\|\boldsymbol{\phi} - \boldsymbol{\phi}_0\|), \tag{8.4}$$

where $J_F(\boldsymbol{\phi}_0)$ is the Jacobian of F computed at $\boldsymbol{\phi}_0$ and F is the function $F(\mathbf{x}) = \frac{\mathbf{x}}{\|\mathbf{x}\|}$. The linear transformation which approximates the length normalization function around the i–vector is then:

$$\widehat{F}(\boldsymbol{\phi}) = F(\boldsymbol{\phi}_0) + J_F(\boldsymbol{\phi}_0)(\boldsymbol{\phi} - \boldsymbol{\phi}_0) = \mathbf{v} + \frac{(\mathbf{I} - \mathbf{v}\mathbf{v}^{\mathrm{T}})}{\|\boldsymbol{\phi}_0\|}\boldsymbol{\phi} \tag{8.5}$$

where $\mathbf{v} = \frac{\boldsymbol{\phi}_0}{\|\boldsymbol{\phi}_0\|}$ and $\mathbf{I}$ is the identity matrix.

The extension to the full i–vector posterior consists in computing the first order Taylor expansion of F centered at the posterior distribution mean $\boldsymbol{\phi}_{\mathcal{X}}$, and applying the resulting linear transformation to the i–vector posterior $\mathbf{W}|\mathcal{X} \sim \mathcal{N}(\boldsymbol{\phi}_{\mathcal{X}}, \boldsymbol{\Gamma}_{\mathcal{X}}^{-1})$. The expansion of F around $\boldsymbol{\phi}_{\mathcal{X}}$ is:

$$\widehat{F}(\boldsymbol{\phi}_{\mathcal{X}}) = \mathbf{v}_{\mathcal{X}} + \frac{(\mathbf{I} - \mathbf{v}_{\mathcal{X}}\mathbf{v}_{\mathcal{X}}^{\mathrm{T}})}{\|\boldsymbol{\phi}_{\mathcal{X}}\|}\boldsymbol{\phi}_{\mathcal{X}} = \mathbf{v}_{\mathcal{X}} + \mathbf{A}\boldsymbol{\phi}_{\mathcal{X}} \ , \tag{8.6}$$

where $\mathbf{v}_{\mathcal{X}} = \frac{\boldsymbol{\phi}_{\mathcal{X}}}{\|\boldsymbol{\phi}_{\mathcal{X}}\|}$ and $\mathbf{A} = \frac{(\mathbf{I} - \mathbf{v}_{\mathcal{X}}\mathbf{v}_{\mathcal{X}}^{\mathrm{T}})}{\|\boldsymbol{\phi}_{\mathcal{X}}\|}$. Thus, the transformed distribution is given by:

$$\begin{aligned}\widehat{\mathbf{W}} &\sim \mathcal{N}\left(\widehat{F}(\boldsymbol{\phi}_{\mathcal{X}}), \mathbf{A}\boldsymbol{\Gamma}_{\mathcal{X}}^{-1}\mathbf{A}^{\mathrm{T}}\right) \\ &\sim \mathcal{N}\left(\frac{\boldsymbol{\phi}_{\mathcal{X}}}{\|\boldsymbol{\phi}_{\mathcal{X}}\|}, \frac{1}{\|\boldsymbol{\phi}_{\mathcal{X}}\|^2}(\mathbf{I} - \mathbf{v}_{\mathcal{X}}\mathbf{v}_{\mathcal{X}}^{\mathrm{T}})\boldsymbol{\Gamma}_{\mathcal{X}}^{-1}(\mathbf{I} - \mathbf{v}_{\mathcal{X}}\mathbf{v}_{\mathcal{X}}^{\mathrm{T}})\right) .\end{aligned} \tag{8.7}$$

Expression (8.7) can be further approximated as:

$$\overline{\mathbf{W}} \sim \mathcal{N}\left(\frac{\boldsymbol{\phi}_{\mathcal{X}}}{\|\boldsymbol{\phi}_{\mathcal{X}}\|}, \frac{\boldsymbol{\Gamma}_{\mathcal{X}}^{-1}}{\|\boldsymbol{\phi}_{\mathcal{X}}\|^2}\right) \ . \tag{8.8}$$

In the experimental section, we show that these linearizations of the length normalization are effective. In particular, the approximation (8.8) allows a simplification of (8.7) without incurring any performance degradation. We will refer to (8.7) as “Projected Length Normalization” (FPD1), and to (8.8) as “Length Normalization” (FPD2).

Chapter 9

Discriminative Training of PLDA

In this chapter, we propose to estimate verification scores using a *discriminative model* rather than a generative PLDA model. More specifically, the speaker verification score for a pair of i-vectors is computed using a function having the functional form derived from the standard PLDA model. The parameters of the function, however, are estimated using a discriminative training criterion. We use an objective function that directly addresses the speaker verification task, i.e. the discrimination between “same-speaker” and “different-speaker” trials. In other words, a binary classifier that takes a pair of i-vectors as an input, is trained to answer the question of whether or not the two i-vectors come from the same speaker. We show that the functional form derived from PLDA can be interpreted as a binary linear classifier in a non-linearly expanded space of i-vector pairs. We have experimented with two discriminative linear classifiers: linear support vector machines (SVM) and logistic regression. The advantage of logistic regression is its probabilistic interpretation: the linear output of this classifier can be directly interpreted as the desired log-likelihood ratio verification score. We will concentrate more on training with logistic regression and we will use the abbreviation DPLDA (Discrminative PLDA) for such systems later in Chapter 10.

There has been previous work on discriminative training for speaker recognition, such as GMM-SVM [Campbell et al., 2006]. This and similar approaches, however, do not directly address the objective of discriminating between same-speaker and different-speaker trials. Instead, SVMs are trained as discriminative models representing each target speaker. As a consequence, this approach cannot fully benefit from discriminative training, as there is a very limited number of positive examples (usually only one enrollment segment) available for training of each model. In contrast, in our approach, a model is trained using a large number of positive and negative examples, each of which is one of many possible same-speaker or different-speaker trials that can be constructed from the training segments.

The very same idea of discriminatively training a PLDA-like model for speaker verification was originally proposed in [Brümmer, 2006] and some initial work has been done in [Burget et al., 2008]. At that time, however, speaker factors extracted using Joint Factor Analysis (JFA) [Kenny et al., 2007] were used as a suboptimal input for the classifier, and state-of-the-art performance was not achieved.

9.1 Original Model

In order to effectively deploy the discriminative approach to speaker recognition, we need to derive an efficient scheme for obtaining scores for the training examples. We will build our model on previously presented LDA principles and consider a special form of PLDA, a *two-covariance model*, where the simplification is obtained by merging together the residual noise and inter-session components. In this model, both speaker and inter-session variabilities are modeled using across-class and within-class full covariance matrices $\boldsymbol{\Sigma}_{ac}$ and $\boldsymbol{\Sigma}_{wc}$. The two-covariance model is a generative linear-Gaussian model, where latent vectors $\mathbf{y}$ representing speakers (or more generally classes) are assumed to be distributed according to prior distribution

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma}_{ac}). \tag{9.1}$$

For a given speaker represented by a vector $\hat{\mathbf{y}}$, the distribution of i-vectors is assumed to be

$$p(\boldsymbol{\phi}|\hat{\mathbf{y}}) = \mathcal{N}(\boldsymbol{\phi}; \hat{\mathbf{y}}, \boldsymbol{\Sigma}_{wc}). \tag{9.2}$$

The maximum likelihood estimates of the model parameters, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}_{ac}$, and $\boldsymbol{\Sigma}_{wc}$, can be obtained by means of EM algorithm similar to the previous sections. Alternatively, if we want to only obtain a reasonable initialization of the parameters for the discriminative training, the parameters can be directly estimated on the training data as for standard LDA. The training data (i-vectors) come from a database comprising recordings of many speakers (to capture across-class variability), each recorded in several sessions (to capture within-class variability).

9.2 Verification Score of a Trial

To obtain an effective way of scoring, we will consider a trial to be composed only by two i–vectors ($\boldsymbol{\phi}_1$, $\boldsymbol{\phi}_2$). Note, that multi-session scoring, when more i–vectors are available for enroll or test or both, can be easily achieved by averaging the corresponding i–vectors and using the resulting means as single i–vectors. The averaging of i–vectors does not cause any significant problems or deterioration of the performance [Villalba et al., 2013] and in fact is widely used in the community.

We will follow the same steps as in Section 6.2.1 but with the constraint of a single i–vector per enroll and test parts of the evaluation trial. In the case of a same-speaker trial (hypothesis H_s), a single vector $\hat{\mathbf{y}}$ representing a particular speaker is generated from the prior $p(\mathbf{y})$, for which both $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ are generated from $p(\boldsymbol{\phi}|\hat{\mathbf{y}})$. For a different-speaker trial (hypothesis H_d), two vectors $\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2$) representing two different speakers are independently generated from $p(\mathbf{y})$. For each, one of the i-vectors $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ is generated. The speaker verification score can be again calculated as a log-likelihood ratio between the two hypotheses H_s and H_d as

$$s \;=\; \log \frac{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|H_s)}{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|H_d)}. \tag{9.3}$$

The joint likelihood of the two independent i–vectors being generated from a particular speaker factor $\hat{\mathbf{y}}$ is the product of two likelihoods:

$$p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2 | \hat{\mathbf{y}}) = p(\boldsymbol{\phi}_1 | \hat{\mathbf{y}})\, p(\boldsymbol{\phi}_2 | \hat{\mathbf{y}}). \tag{9.4}$$

Considering the hypothesis H_s that these two i–vectors can be generated by any speaker common for both of them, we marginalize over all possible speakers:

$$p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2 | H_s) = \int p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2 | \mathbf{y})\, p(\mathbf{y}) \mathrm{d}\mathbf{y}. \tag{9.5}$$

For the different speaker hypothesis H_d, we again marginalize over all possible speakers and compute the likelihood of the i–vectors being generated independently by any two speakers:

$$\begin{aligned} p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2 | H_d) &= \int p(\boldsymbol{\phi}_1 | \mathbf{y_1})\, p(\mathbf{y}_1)\, \mathrm{d}\mathbf{y}_1 \int p(\boldsymbol{\phi}_2 | \mathbf{y_2})\, p(\mathbf{y}_2)\, \mathrm{d}\mathbf{y}_2, \\ &= p(\boldsymbol{\phi}_1)\, p(\boldsymbol{\phi}_2). \end{aligned} \tag{9.6}$$

Plugging the conditional likelihoods (9.5) and (9.6) into the log-likelihood ration (9.3) we obtain

$$s = \log \frac{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2 | H_s)}{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2 | H_d)} \tag{9.7}$$

$$= \log \frac{\int p(\boldsymbol{\phi}_1 | \mathbf{y}) p(\boldsymbol{\phi}_2 | \mathbf{y}) p(\mathbf{y}) d\mathbf{y}}{p(\boldsymbol{\phi}_1) p(\boldsymbol{\phi}_2)}. \tag{9.8}$$

The integrals, which can be interpreted as convolutions of Gaussians, can be evaluated analytically giving

$$\begin{aligned} s &= \log \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \end{bmatrix} ; \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right) \\ &- \log \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \end{bmatrix} ; \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right), \end{aligned} \tag{9.9}$$

where the total covariance matrix is given as $\boldsymbol{\Sigma}_{tot} = \boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc}$. By expanding the log of Gaussian distributions and simplifying the final expression, we obtain

$$\begin{aligned} s &= \boldsymbol{\phi}_1^T \boldsymbol{\Lambda} \boldsymbol{\phi}_2 + \boldsymbol{\phi}_2^T \boldsymbol{\Lambda} \boldsymbol{\phi}_1 + \boldsymbol{\phi}_1^T \boldsymbol{\Gamma} \boldsymbol{\phi}_1 + \boldsymbol{\phi}_2^T \boldsymbol{\Gamma} \boldsymbol{\phi}_2 \\ &+ (\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2)^T \mathbf{c} + k, \end{aligned} \tag{9.10}$$

where

$$\begin{aligned} \boldsymbol{\Gamma} &= -\frac{1}{4}(\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1} - \frac{1}{4}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{2}\boldsymbol{\Sigma}_{tot}^{-1} \\ \boldsymbol{\Lambda} &= -\frac{1}{4}(\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1} + \frac{1}{4}\boldsymbol{\Sigma}_{wc}^{-1} \\ \mathbf{c} &= ((\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1} - \boldsymbol{\Sigma}_{tot}^{-1})\boldsymbol{\mu} \end{aligned}$$

$$\begin{aligned} k &= \log|\boldsymbol{\Sigma}_{tot}| - \frac{1}{2}\log|\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac}| - \frac{1}{2}\log|\boldsymbol{\Sigma}_{wc}| \\ &\quad + \boldsymbol{\mu}^T(\boldsymbol{\Sigma}_{tot}^{-1} - (\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1})\boldsymbol{\mu}. \end{aligned} \tag{9.11}$$

We recall that the computation of a bilinear form $\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{y}$ can be expressed in terms of the Frobenius inner product as $\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{y} = \langle \mathbf{A}, \mathbf{x}\mathbf{y}^{\mathrm{T}} \rangle = \mathrm{vec}(\mathbf{A})^{\mathrm{T}}\mathrm{vec}(\mathbf{x}\mathbf{y}^{\mathrm{T}})$, where $\mathrm{vec}(\cdot)$ stacks the columns of a matrix into a vector. Therefore, the log-likelihood ratio score can be written as a dot product of a vector of weights $\mathbf{w}^{\mathrm{T}}$, and an expanded vector $\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ representing a trial:

$$\begin{aligned} s &= \mathbf{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) \\ &= \begin{bmatrix} \mathrm{vec}(\boldsymbol{\Lambda}) \\ \mathrm{vec}(\boldsymbol{\Gamma}) \\ \mathbf{c} \\ k \end{bmatrix}^T \begin{bmatrix} \mathrm{vec}(\boldsymbol{\phi}_1\boldsymbol{\phi}_2^{\mathrm{T}} + \boldsymbol{\phi}_2\boldsymbol{\phi}_1^{\mathrm{T}}) \\ \mathrm{vec}(\boldsymbol{\phi}_1\boldsymbol{\phi}_1^{\mathrm{T}} + \boldsymbol{\phi}_2\boldsymbol{\phi}_2^{\mathrm{T}}) \\ \boldsymbol{\phi}_1 + \boldsymbol{\phi}_2 \\ 1 \end{bmatrix}. \end{aligned} \tag{9.12}$$

Hence, we have obtained a generative generalized linear classifier [Bishop, 2006], where the probability for a same-speaker trial can be computed from the log-likelihood ratio score using the sigmoid activation function as

$$p(H_s|\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \sigma\left(\log\frac{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|H_s)}{1 - p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|H_s)} + \log\frac{p(H_s)}{1 - p(H_s)}\right) = \sigma(s + \mathrm{logit}(p(H_s))). \tag{9.13}$$

Adding the $\mathrm{logit}(p(H_s))$ score, which adjusts the constant k in the vector of weights, allows for setting different priors for both hypotheses.

9.3 Discriminative Classifiers

In this section, we describe how we train the weights $\mathbf{w}$ directly, in order to discriminate between same-speaker and different-speaker trials, without having to explicitly model the distributions of i-vectors. To represent a trial, we keep the same expansion $\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ as defined in (9.12). Hence, we reuse the functional form for computing verification scores that provided excellent results with generative PLDA. We consider two standard discriminative linear classifiers, namely logistic regression and SVMs.

9.3.1 Logistic Regression

The set of training examples $\mathbf{r}_1 \ldots \mathbf{r}_{|\mathcal{T}|} \in \mathcal{T}$, which we continue referring to as training trials, comprises both different-speaker and same-speaker trials. By trial $\mathbf{r}$ we understand a combination of two i–vectors $\mathbf{r} = (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$. By introducing the variable for trial, our score for a particular trial becomes $s_{\mathbf{r}} = \mathbf{w}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{r}) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$. Let us also define the coding scheme $t \in \{-1, 1\}$ to represent labels for the different-speaker, and same-speaker trials, respectively. Assigning each trial a log-likelihood ratio $s_{\mathbf{r}}$ and the correct label $t_{\mathbf{r}}$, the log probability of recognizing the trial correctly can be expressed as

$$\log p(t_{\mathbf{r}}|\mathbf{r}) = -\log(1 + \exp(-s_{\mathbf{r}}t_{\mathbf{r}})). \tag{9.14}$$

This is easy to see from equation (9.13) and recalling that $\sigma(-s) = 1 - \sigma(s)$. In the case of logistic regression, the objective function to maximize with respect to the optimized parameters $\mathbf{w}$ is the log posterior probability of correct labeling of all training examples, i.e. the sum of expressions (9.14) evaluated for all training trials.

$$\begin{aligned} \mathcal{Q} &= \sum_{\mathbf{r}\in\mathcal{T}} \log p(t_{\mathbf{r}}|s_{\mathbf{r}}(\mathbf{w})) && (9.15)\\ &= \sum_{\mathbf{r}\in\mathcal{T}} -\log\left(1 + \exp(-t_{\mathbf{r}}s_{\mathbf{r}}(\mathbf{w}))\right) && (9.16)\end{aligned}$$

Equivalently, this can be expressed by minimizing the cross-entropy error function, which is a sum over all training trials

$$E(\mathbf{w}) = \sum_{\mathbf{r}\in\mathcal{T}} \alpha_{\mathbf{r}} E_{LR}(t_{\mathbf{r}}s_{\mathbf{r}}) \tag{9.17}$$

where the logistic regression loss function

$$E_{LR}(t_{\mathbf{r}}s_{\mathbf{r}}) = \log(1 + \exp(-t_{\mathbf{r}}s_{\mathbf{r}})) \tag{9.18}$$

is simply the negative log probability (9.14) of correctly recognizing a trial.

To control over-fitting to training data and to keep the optimized parameters from reaching large values, we can introduce a regularization by adding a penalty term to the error function. The simplest form of the regularization penalty is the sum of squares of all parameters, leading to a modified error function

$$\tilde{E}(\mathbf{w}) = \sum_{\mathbf{r}\in\mathcal{T}} \alpha_{\mathbf{r}} E_{LR}(t_{\mathbf{r}}s_{\mathbf{r}}) + \frac{\lambda}{2}\|\mathbf{w}\|^2, \tag{9.19}$$

where $\|\mathbf{w}\|^2 = \mathbf{w}^{\mathrm{T}}\mathbf{w}$ and the coefficient λ is a constant controlling the tradeoff between the error function and the regularizer. This L_2 regularizer can be extended by incorporating a prior knowledge of the parameters $\mathbf{w}$ and therefore allow it to limit the distance of the optimized parameters from some particular offset (for example the parameters estimated from the generative model). The error function then takes the form of

$$\tilde{E}(\mathbf{w}) = \sum_{\mathbf{r}\in\mathcal{T}} \alpha_{\mathbf{r}} E_{LR}(t_{\mathbf{r}}s_{\mathbf{r}}) + \frac{\lambda}{2}\|\mathbf{w} - \hat{\mathbf{w}}\|^2. \tag{9.20}$$

This regularization can be seen as imposing an isotropic Gaussian prior on the parameters [Bishop, 2006]. The $\hat{\mathbf{w}}$ defines the mean of the isotropic Gaussian prior and the regularization constant λ can be seen as a parameter to control the variance of this prior.

The coefficients $\alpha_{\mathbf{r}}$ allow us to weight individual trials. When set to zero, it can be used to “turn off” some unwanted trials – for example same i–vector trials or cross-gender trials. We use these coefficients also to assign different weights to same-speaker and different-speaker trials. This allows us to select a particular operating point, around which we want to optimize the performance of our system without relying on the proportion of same- and different-speaker trials in the training set. The advantage of using the cross-entropy objective for training is that it reflects performance of the system over a wide range of operating points (around the selected one). We can show that by setting the α coefficients proportional to the number of same- ($|\mathcal{T}_1|$) and different-speaker trials ($|\mathcal{T}_2|$) as $\frac{1}{2\log(2)|\mathcal{T}_1|}$ and $\frac{1}{2\log(2)|\mathcal{T}_2|}$, our error function without regularization becomes

$$\begin{aligned} E_{\mathcal{T}}(\mathbf{w}) &= \frac{1}{2\log(2)}\left(\frac{1}{|\mathcal{T}_1|}\sum_{\mathbf{r}\in\mathcal{T}}\log(1+\exp(s_{\mathbf{r}}(\mathbf{w}))) + \frac{1}{|\mathcal{T}_2|}\sum_{\mathbf{r}\in\mathcal{T}}\log(1+\exp(s_{\mathbf{r}}(\mathbf{w})))\right) \quad (9.21) \\ &= C_{llr,\mathbf{w}}(\mathcal{T}), \end{aligned}$$

which is the C_{llr} performance measure for the speaker verification task as defined in [Brümmer and du Preez, 2006]. This probabilistic behavior of the logistic regression classifier is one of its advantages against the SVM as it trains the weights so that the score $s_{\mathbf{r}} = \mathbf{w}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{r}) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ can be interpreted as the log-likelihood ratio between hypotheses H_s and H_d, and therefore, the calibration step is not so necessary.

9.3.2 Gradient Evaluation

In order to numerically optimize the parameters $\mathbf{w}$ of the classifier, we want to evaluate the gradient of the error function

$$\nabla E(w) = \sum_{\mathbf{r}\in\mathcal{T}} \alpha_{\mathbf{r}} \frac{\partial E(t_{\mathbf{r}}s_{\mathbf{r}})}{\partial s_{\mathbf{r}}}\frac{\partial s_{\mathbf{r}}}{\partial \mathbf{w}} + \lambda\mathbf{w}, \quad (9.22)$$

where the derivation of the loss function $E(t_{\mathbf{r}}s_{\mathbf{r}})$, w.r.t. score $s_{\mathbf{r}}$, depends on the particular choice of the loss function. For the logistic regression loss function, it is

$$\frac{\partial E_{LR}(t_{\mathbf{r}}s_{\mathbf{r}})}{\partial s_{\mathbf{r}}} = -t_{\mathbf{r}}\sigma(-t_{\mathbf{r}}s_{\mathbf{r}}). \quad (9.23)$$

Finally, the derivation of the score w.r.t. the classifier parameters just gives the expanded trial vector

$$\frac{\partial s}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}}\mathbf{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2). \quad (9.24)$$

9.3.3 Efficient Score and Gradient Evaluation

Given a trained classifier, we can obtain a verification score for a trial by forming the expanded vector $\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ and computing the dot product (9.12). However, as we have already seen, the same score can be obtained using the two original i-vectors $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2$ and using formula (9.10), which is both memory and computationally efficient. Now, consider two sets of i-vectors stored as columns of matrices $\boldsymbol{\Phi}_e$ and $\boldsymbol{\Phi}_t$. For illustration, let us call these sets enrollment and test trials, although they play symmetrical roles in our scoring scheme. We can efficiently score each enrollment trial against each test trial and obtain the full matrix of scores as

$$\begin{aligned}\mathbf{S} \;=\;& 2\boldsymbol{\Phi}_e^T\boldsymbol{\Lambda}\boldsymbol{\Phi}_t\\ &+((\boldsymbol{\Phi}_e^T\boldsymbol{\Gamma})\circ\boldsymbol{\Phi}_e^{\mathrm{T}})\mathbf{1}\mathbf{1}^{\mathrm{T}}+\mathbf{1}\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Phi}_t\circ(\boldsymbol{\Gamma}\boldsymbol{\Phi}_t))\\ &+\boldsymbol{\Phi}_e^{\mathrm{T}}\mathbf{c}\mathbf{1}^T+\mathbf{1}\mathbf{c}^{\mathrm{T}}\boldsymbol{\Phi}_t^{\mathrm{T}}+k,\end{aligned} \tag{9.25}$$

where $\circ$ denotes the Hadamard, or “entrywise” product. Terms $\mathbf{1}$ and $\mathbf{1}\mathbf{1}^{\mathrm{T}}$ represent the vector and the matrix of ones, respectively. Similarly, the naïve way of evaluating the gradient would be to explicitly expand every training trial and then to apply equations (9.22) to (9.24). However, again taking into account the functional form for computing scores (9.10), the gradient can be evaluated much more efficiently without any need for explicit trial expansion. Let all the i-vectors, which we have available for training, be stored in columns of a matrix $\boldsymbol{\Phi}$. Now consider forming a training trial using every possible pair of i-vectors from the matrix. Let s_{ij} be the score for the trial formed by the i-th and j-th columns of $\boldsymbol{\Phi}$ calculated using the parameters $\mathbf{w}$ for which we wish to evaluate the gradient. Let t_{ij} and α_{ij} be the corresponding label and trial weight, respectively. Further, let d_{ij} be the corresponding derivation of loss function $E(t_{ij}s_{ij})$ w.r.t. the score s_{ij} given in (9.23) or (9.29) depending on the loss function used. The gradient can now be efficiently evaluated as

$$\nabla E(\mathbf{w})=\begin{bmatrix}\nabla_\Lambda L\\ \nabla_\Gamma L\\ \nabla_c L\\ \nabla_k L\end{bmatrix}=\begin{bmatrix}2\cdot\mathrm{vec}\left(\boldsymbol{\Phi}\mathbf{G}\boldsymbol{\Phi}^{\mathrm{T}}\right)\\ 2\cdot\mathrm{vec}\left(\boldsymbol{\Phi}[\boldsymbol{\Phi}^{\mathrm{T}}\circ(\mathbf{G}\mathbf{1}\mathbf{1}^{\mathrm{T}})]\right)\\ 2\cdot\mathbf{1}^{\mathrm{T}}[\boldsymbol{\Phi}^{\mathrm{T}}\circ(\mathbf{G}\mathbf{1}\mathbf{1}^{\mathrm{T}})]\\ \mathbf{1}^{\mathrm{T}}\mathbf{G}\mathbf{1}\end{bmatrix}+\lambda\mathbf{w}, \tag{9.26}$$

where elements of matrix $\mathbf{G}$ are $g_{ij}=d_{ij}\cdot\alpha_{ij}$. The form of the gradient, allowing us to split the training i–vectors into two different sets of “enrollment” i–vectors $\boldsymbol{\Phi}_e$ and “test” i–vectors $\boldsymbol{\Phi}_t$, can be written as

$$\nabla E(\mathbf{w})=\begin{bmatrix}\nabla_\Lambda L\\ \nabla_\Gamma L\\ \nabla_c L\\ \nabla_k L\end{bmatrix}=\begin{bmatrix}\mathrm{vec}\left(\boldsymbol{\Phi}_e\mathbf{G}\boldsymbol{\Phi}_t^T+\boldsymbol{\Phi}_t\mathbf{G}\boldsymbol{\Phi}_e^T\right)\\ \mathrm{vec}\left(\boldsymbol{\Phi}_e[\boldsymbol{\Phi}_e^T\circ(\mathbf{G}\mathbf{1}\mathbf{1}^T)]+[(\mathbf{1}\mathbf{1}^T\mathbf{G})\circ\boldsymbol{\Phi}_t]\boldsymbol{\Phi}_t^T\right)\\ \mathrm{vec}\left(\mathbf{1}^T[\boldsymbol{\Phi}_e^T\circ(\mathbf{G}\mathbf{1}\mathbf{1}^T)]+[(\mathbf{1}\mathbf{1}^T\mathbf{G})\circ\boldsymbol{\Phi}_t]\mathbf{1}^T\right)\\ \mathbf{1}^T\mathbf{G}\mathbf{1}\end{bmatrix}+\lambda\mathbf{w}. \tag{9.27}$$

9.3.4 Support Vector Machines

For the completeness, we will show how to train SVMs using the proposed scheme. The detailed analysis of this problem for the SVM is given in [Cumani et al., 2013]. We will focus on putting the SVM approach to the relation with previously described LR. It is straightforward to find the relation between SVM and LR as we can see both approaches as a particular instance of the unconstrained convex regularized risk minimization problem. To formulate the classifier as the SVM, we just need to insert a hinge loss function E_{L1} instead of logistic regression loss function (9.18) into (9.20) and minimize it with respect to $\mathbf{w}$. The hinge loss function is defined as

$$E_{L1}(t_{\mathbf{r}}s_{\mathbf{r}})=\max(0,1-t_{\mathbf{r}}s_{\mathbf{r}}). \tag{9.28}$$

and its derivative with respect to scores $s_{\mathbf{r}}$ is defined as

$$\frac{\partial E_{L1}(t_{\mathbf{r}}s_{\mathbf{r}})}{\partial s_{\mathbf{r}}}=\begin{cases}0 & \text{if} \quad t_{\mathbf{r}}s_{\mathbf{r}}\geq 1\\ -t_{\mathbf{r}} & \text{otherwise.}\end{cases} \tag{9.29}$$

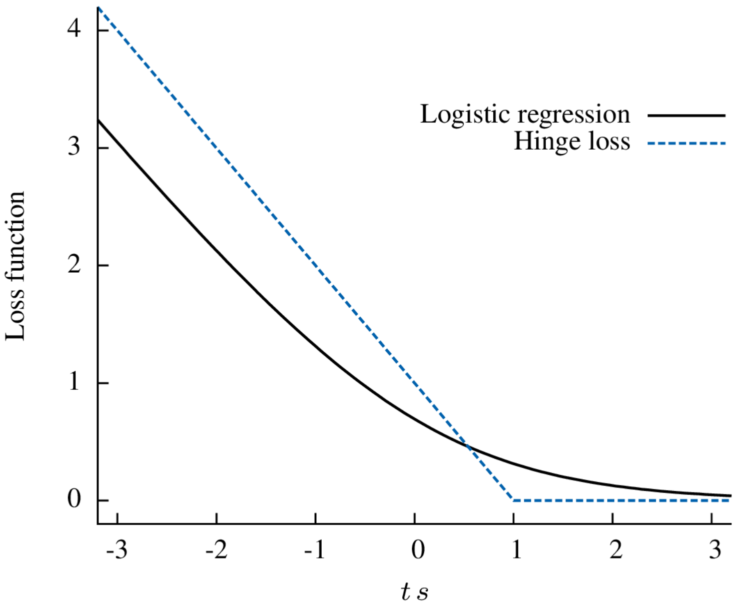


Figure 9.1: Loss functions for logistic regression and SVM.

This way, we obtain an SVM, which is a classifier optimizing the separation margin between the classes, whereas LR minimizes the cross-entropy error function. Alternatively, one can see the hinge loss function as a piecewise approximation to the logistic regression loss function. Therefore, one can assume that the score $s = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{r})$ obtained from an SVM classifier will still be a reasonable approximation to the log-likelihood ratio (after a linear calibration). Both loss functions are shown in Figure 9.1.

SVM Training

As an SVM in its basic definition is a linear classifier, it can extend it to a non-linear classifier by applying the so called "kernel trick"[Boser et al., 1992], where every dot product between two examples is replaced by a non-linear kernel function. In our case, every training example is represented by a non-linear expansion of two i–vectors forming trial (see (9.12)). We will show that a second degree polynomial kernel

$$K(\mathbf{r}_1, \mathbf{r}_2) = (\mathbf{r}_1^{\mathrm{T}} \mathbf{r}_2 + 1)^2, \tag{9.30}$$

where $\mathbf{r}_1 = [\boldsymbol{\phi}_a \, \boldsymbol{\phi}_b]$, $\mathbf{r}_2 = [\boldsymbol{\phi}_w \, \boldsymbol{\phi}_z]$ represent two different speaker verification trials, is equivalent to the dot product between two training examples. The kernel

$$\begin{aligned} K(\mathbf{r}_1, \mathbf{r}_2) &= K([\boldsymbol{\phi}_a \, \boldsymbol{\phi}_b] \, , [\boldsymbol{\phi}_w \, \boldsymbol{\phi}_z]) \\ &= (\boldsymbol{\phi}_a^{\mathrm{T}} \boldsymbol{\phi}_w + \boldsymbol{\phi}_b^{\mathrm{T}} \boldsymbol{\phi}_z + 1)^2 \end{aligned} \tag{9.31}$$

can be rewritten as:

$$\begin{aligned}K(\mathbf{r}_1, \mathbf{r}_2) &= \boldsymbol{\phi}_a^{\mathrm{T}} \boldsymbol{\phi}_w \boldsymbol{\phi}_w^{\mathrm{T}} \boldsymbol{\phi}_a + \boldsymbol{\phi}_b^{\mathrm{T}} \boldsymbol{\phi}_z \boldsymbol{\phi}_z^{\mathrm{T}} \boldsymbol{\phi}_b \\ &\quad + 2\boldsymbol{\phi}_a^{\mathrm{T}} \boldsymbol{\phi}_w \boldsymbol{\phi}_z^{\mathrm{T}} \boldsymbol{\phi}_b + 2\boldsymbol{\phi}_a^{\mathrm{T}} \boldsymbol{\phi}_w + 2\boldsymbol{\phi}_b^{\mathrm{T}} \boldsymbol{\phi}_z + 1 \\ &= \left\langle \boldsymbol{\phi}_a \boldsymbol{\phi}_a^{\mathrm{T}}, \boldsymbol{\phi}_w \boldsymbol{\phi}_w^{\mathrm{T}} \right\rangle + \left\langle \boldsymbol{\phi}_b \boldsymbol{\phi}_b^{\mathrm{T}}, \boldsymbol{\phi}_z \boldsymbol{\phi}_z^{\mathrm{T}} \right\rangle \\ &\quad + 2\left\langle \boldsymbol{\phi}_a \boldsymbol{\phi}_b^{\mathrm{T}}, \boldsymbol{\phi}_w \boldsymbol{\phi}_z^{\mathrm{T}} \right\rangle + 2\boldsymbol{\phi}_a^{\mathrm{T}} \boldsymbol{\phi}_w + 2\boldsymbol{\phi}_b^{\mathrm{T}} \boldsymbol{\phi}_z + 1 \\ &= \left\langle [\boldsymbol{\phi}_a \, \boldsymbol{\phi}_b \, 1] \, [\boldsymbol{\phi}_a \, \boldsymbol{\phi}_b \, 1]^{\mathrm{T}}, [\boldsymbol{\phi}_w \, \boldsymbol{\phi}_z \, 1] \, [\boldsymbol{\phi}_w \, \boldsymbol{\phi}_z \, 1]^{\mathrm{T}} \right\rangle, \end{aligned} \tag{9.32, 9.33}$$

where $\langle A, B \rangle$ is a dot product between two matrices. We can observe a feature mapping in the structure of the expanded kernel function:

$$\tilde{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \operatorname{vec}([\boldsymbol{\phi}_1 \, \boldsymbol{\phi}_2 \, 1] \, [\boldsymbol{\phi}_1 \, \boldsymbol{\phi}_2 \, 1]^{\mathrm{T}}) \sim \begin{bmatrix} \operatorname{vec}(\boldsymbol{\phi}_1 \boldsymbol{\phi}_2^{\mathrm{T}}) \\ \operatorname{vec}(\boldsymbol{\phi}_2 \boldsymbol{\phi}_1^{\mathrm{T}}) \\ \operatorname{vec}(\boldsymbol{\phi}_1 \boldsymbol{\phi}_1^{\mathrm{T}}) \\ \operatorname{vec}(\boldsymbol{\phi}_2 \boldsymbol{\phi}_2^{\mathrm{T}}) \\ \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \\ \boldsymbol{\phi}_2 \\ 1 \end{bmatrix}, \tag{9.34}$$

where $\sim$ denotes the equivalence of vectors ignoring the order of their elements. We can now see the kernel $K(\mathbf{r}_1, \mathbf{r}_2)$ as the dot product of two expansions:

$$K(\mathbf{r}_1, \mathbf{r}_2) = \tilde{\varphi}(\boldsymbol{\phi}_a \boldsymbol{\phi}_b)^{\mathrm{T}} \tilde{\varphi}(\boldsymbol{\phi}_w \boldsymbol{\phi}_z). \tag{9.35}$$

Taking the likelihood in (9.10) and halving its unknown parameter $\mathbf{c}$ as $\tilde{\mathbf{c}} = \mathbf{c}/2$, so that the linear term of the log-likelihood becomes $2\tilde{\mathbf{c}}(\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2)$, the expansion of the i–vectors given in (9.12) becomes

$$\varphi(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \begin{bmatrix} \operatorname{vec}(\boldsymbol{\phi}_1 \boldsymbol{\phi}_2^{\mathrm{T}} + \boldsymbol{\phi}_2 \boldsymbol{\phi}_1^{\mathrm{T}}) \\ \operatorname{vec}(\boldsymbol{\phi}_1 \boldsymbol{\phi}_1^{\mathrm{T}} + \boldsymbol{\phi}_2 \boldsymbol{\phi}_2^{\mathrm{T}}) \\ 2(\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2) \\ 1 \end{bmatrix} \tag{9.36}$$

and it is easy to verify that the two expansions:

$$\varphi(\boldsymbol{\phi}_a \boldsymbol{\phi}_b)^{\mathrm{T}} \varphi(\boldsymbol{\phi}_w \boldsymbol{\phi}_z) = \tilde{\varphi}(\boldsymbol{\phi}_a \boldsymbol{\phi}_b)^{\mathrm{T}} \tilde{\varphi}(\boldsymbol{\phi}_w \boldsymbol{\phi}_z) \tag{9.37}$$

are equivalent and therefore correspond to the same kernel.

Having defined the kernel function, we can now train the SVM by solving either dual or primal optimization problem. The SVM classifiers are often trained using a dual problem, where the Gram matrix of all dot products between every pair of training samples has to be evaluated. In our case, the trials are formed by pairs of i–vectors which results in a complexity $O(|\mathcal{T}|^2)$ or $O(n^4)$ with n being the number of all i–vectors in the training set. Since the size of the training set can be easily tens of thousands of i–vectors, the resulting Gram matrix would be unacceptably large. Therefore, it is better to tackle the problem by formulating an efficient evaluation of the loss function gradient and using a general solver to solve the primal problem as shown in [Cumani et al., 2013].

Recently, an optimization for the pairwise SVM training was proposed in [Cumani and Laface, 2014b], where discarding the non-contributing training pairs leads to a substantial reduction of number of support vectors. Authors show that the number of support vectors can grow linearly with the number of speakers, instead of quadratically with the number of training pairs, which allows for using this technique for larger training datasets, and also to use the dual formulation for SVM training.

9.3.5 Numerical Optimization

The experiments conducted with the logistic regression classifier are using numerical optimization methods based on the iterative “trust-region Newton-conjugate-gradient” method described in [Lin et al., 2008, Nocedal and Wright, 2006] and “Limited-memory BFGS” (L-BFGS) [Nocedal, 1980, Nocedal and Wright, 2006]. The two methods provide the same results, while the first uses a conjungate-gradient for the Hessian inversion and the latter keeps updating certain amount of vectors which implicitly represent the inverse Hessian approximation.

Trust-region Newton-conjugate Gradient Method

The “trust region” corresponds to the spherical region built around the current guess of the solution of the optimization problem where the approximate model is built. The idea of the algorithm is to “trust” the model only in this region, which corresponds to the fact that the general nonlinear approximations, e.g. quadratic approximation fit the original function only locally. This region is adjusted with the consecutive iterations – it can be enlarged if the approximate model fits the problem well, otherwise it is reduced.

The core of the optimization is based on the Newton’s optimization method, where the gradient and an inverse of the Hessian are used for faster convergence. As the evaluation of the Hessian or its inversion would in our case be very memory and computationally expensive operations, the conjugate-gradient method is used for the inversion and the update step is computed by means of Hessian-vector multiplication. These properties require a particular implementation of the algorithm to provide a function which efficiently computes only such multiplication without computing the whole Hessian matrix. An effective way to compute this second order Hessian-vector product is a “complex step differentiation”[Shampine, 2007], however, due to the numerical requirements and code optimization, we resorted to using a real-step numerical approximation, where the product is expressed in terms of two very close gradient vectors.

9.3.6 Limited-memory BFGS Method

The “BFGS” in the name of the method stands for the names of the four people who independently discovered it in 1970: Broyden, Fletcher, Goldfarb and Shanno. Later, the method was modified by Nocedal [Nocedal, 1980] who introduced a memory efficient variant suitable for larger problems. The L-BFGS method for unconstrained optimization is usually implemented as a line search method, where BFGS approximation to the inverse of the Hessian is used to obtain the search direction in every iteration. It was shown [Nocedal and Nash, 1991, Liu et al., 1989] that methods based on this scheme of iterative updating are effective for large scale nonlinear problems and compete well with approximate iterative Newton methods.

Implementations of the Optimizers

The implementation of the Trust-region Newton method was taken from the code developed for the BOSARIS toolkit [Brümmer and de Villiers, 2010]. This toolkit allows for building various objective functions by a composition of the basic multivariate and twice differentiable functions. Each atomic function has to be able to compute the function value and also, if requested, provide the first and second-order partial derivatives in order to allow for a back-propagation of the gradient for the evaluation of the complete composite gradient.

During the time, we also used the L-BFGS as it was readily available in the SciPy library for the Python programming language. In fact, the SciPy optimization package is a wrapper around the Nocedal's implementation in Fortran.

Chapter 10

Experimental Results

This chapter will present results obtained with the presented techniques on various datasets. First, to put the techniques into the historical context, we will present a short description and performance comparison of the past state-of-the-art techniques on a common SRE 2010 dataset in Section 10.1. In Section 10.2, we will take the standard PLDA without any i–vector normalization as a baseline and show (still on SRE 2010 dataset) the effects of discriminatively trained PLDA and i–vector length normalization. Next, we will concentrate on the analysis of PLDA and DPLDA in diverse acoustic conditions (Section 10.3). Finally, we will compare all presented PLDA techniques on NIST SRE 2012 dataset (Section 10.4). The superiority of the full-posterior PLDA for short segments, where the uncertainty of extracted i–vectors is high, will be demonstrated on modified NIST SRE 2010 datasets. The modifications consists only in truncating the enrollment and test segments into various lengths in order to simulate the scenario when low amount of data is available for i–vector estimation.

10.1 Overview of Techniques

We will present a comparison of the PLDA with previous techniques considered as state-of-the-art before introduction of i–vectors and PLDA. All systems are built on top of comparable training datasets. There can be only minor changes in the training lists introduced by the corrections and small modifications during the years of development and refinement of the database labels.

10.1.1 Common Setup

To present a fair comparison of the SRE techniques, we built all of the corresponding systems on top of the architecture designed for ABC submission for the NIST SRE 2010 evaluations [Brümmer et al., 2010b]. All systems with exception of the JFA system share the same feature extraction, voice activity detection and UBM. For the JFA system, gender-dependent UBM was used.

Feature Extraction

We use 19-dimensional MFCC coefficients + log energy with analysis window of 20 ms and a shift of 10 ms. Corresponding delta and double delta coefficients are computed resulting in 60-dimensional feature vectors.

After removing the silence, we apply short-time Gaussianization with window of 300 frames. On the border sides of the feature vector sequence, only 150 frames are used. We let the window grow or shrink in the beginning or in the end of the sequence.

Voice Activity Detection

Speech/silence segmentation is based on a hybrid of Artificial Neural Networks and Hidden Markov Model [Schwarz et al., 2006]. It is used as a phoneme recognizer trained on the SPEECHDAT Hungarian database [Matějka et al., 2006]. The outputs are phonemes clustered into two classes: "speech" (all speech phonemes) and "silence" (all models for silence). The resulting clusters of speech and silence are then post-processed using a relative average energy thresholding. The rules are invented heuristically in order to overcome the problems of false speech detection and in case of 2-channel data to avoid cross-talks. The process is as follows:

1. If the average energy of the "speech" segment is lower than the maximum energy in the whole utterance minus 30 dB, then the segment is labeled as "silence". This step is done for every utterance.

2. For the 2-channel files: If the energy in the other channel is greater than the maximum energy minus 3 dB in the channel which is being processed, the segment is also labeled as "silence".

It should be noted that for data where NIST provided ASR transcripts (interview data) only rule 1 of the post-processing was applied. The cross-talks were eliminated by marking all of the segments where interviewer was speaking as "silence". Note that we will not show results on interview data in this section, but the very same VAD will be used also in the other experiments, where we test on all conditions, which include also these data.

UBM and I–vector Extractor

A gender-independent universal background model is represented by a full-covariance, 2048-component GMM. The UBM and gender-dependent i–vector extractors were trained on NIST SRE 2004, 2005 and 2006, Switchboard II Phases 2 and 3, and Switchboard Cellular Parts 1 and 2 and Fisher English Parts 1 and 2. The variance flooring was used in each iteration of EM algorithm during the UBM training. We extracted 400-dimensional i–vectors for the cosine distance and PLDA systems. The analysis regarding the use of diagonal and full-covariance UBM was performed in [Matějka et al., 2011].

Data for Score Normalization

For the systems, where the score normalization is necessary, we were using NIST SRE 2004, 2005 and 2006 data. This data were filtered in such a way that there were at least five sessions for each speaker.

10.1.2 Individual Systems

We will briefly describe individual compared systems. Some of these systems were part of the ABC NIST SRE 2010 and more variants were trained to suit particular test conditions. We shall concentrate only on the most common extended telephone-telephone condition (condition 5) and therefore describe only telephone systems. Generally the modification for other conditions is based only on a different selection of training data. For the details, see [Brümmer et al., 2010b].

Relevance MAP and Eigenchannel Adaptation

Scores from both systems were normalized by zt-norm using 200 speakers for the z-norm and t-norm segments. In case of the eigenchannel adaptation, 50 eigenchannels were trained on the same data as for the score normalization.

JFA

A gender dependent telephone system was trained using data from NIST 2004, 2005, 2006, Switchboard II Phases 2 and 3, and Switchboard Cellular Parts 1 and 2. The numbers of eigenvoices and eigenchannels we 300 and 100, respectively. Both eigenchannel and eigenvoice matrices were randomly initialized and then trained with 10 EM iterations of maximum likelihood followed by the minimum divergence step. The training was done always separately for each subspace (eigenvoices and eigenchannels) with the other fixed. No matrix for the residual variability was used.

The scores were normalizes with zt-norm using 200 speakers for the z-norm and t-norm segments.

Cosine Distance

We follow an already described (see Section 5.3) scheme of intersession variability compensation by means of the LDA followed by the WCCN. 400-dimensional i–vectors were reduced by LDA into 200 dimensions. The LDA transformation matrix was trained separately for male and female subset on the same data as i–vector extractor, except the Fisher data that was excluded.

We used simplified symmetrical normalization (s-norm, see Section 1.2.1). Gender dependent s-norm cohort was created by 400 speakers.

PLDA

Standard PLDA models are also gender dependent and trained using 400 dimensional i-vectors extracted from 21663 segments from 1384 female speakers and 16969 segments from 1051 male speakers from NIST SRE 2004, NIST SRE 2005, NIST SRE 2006, Switchboard II Phases 2 and 3, and Switchboard Cellular Parts 1 and 2. The configuration of the PLDA was 90 eigenvoices, eigenchannels are full rank with dimensionality 400. This particular configuration was tuned to give the best results for $\mathrm{DCF}_{\mathrm{new}}$, while having sub-optimal performance at the $\mathrm{DCF}_{\mathrm{old}}$ and EER. In the case of PLDA, no score normalization was used.

10.1.3 Results

In Figure 10.1, we can observe the evolution of the SRE systems. Clearly, the introduction of the channel adaptation has dramatically improved the performance, especially when the system was evaluated on data coming from different collection or simply containing channel effects not present during the UBM training.

JFA was another milestone, which greatly improved the performance at the time when it was introduced. Surprisingly, the effect is not so big on the NIST SRE 2010. However this technique led to the introduction of i–vectors and we can observe another substantial gain in the performance with the cosine distance scoring of i–vectors.

If we compare PLDA with the cosine distance scoring, we do not see much of a difference between the two systems. In fact, the cosine distance scoring is better on the low miss-rate region of the DET curve. However, this situation has changed in favor of PLDA after applying length normalization.

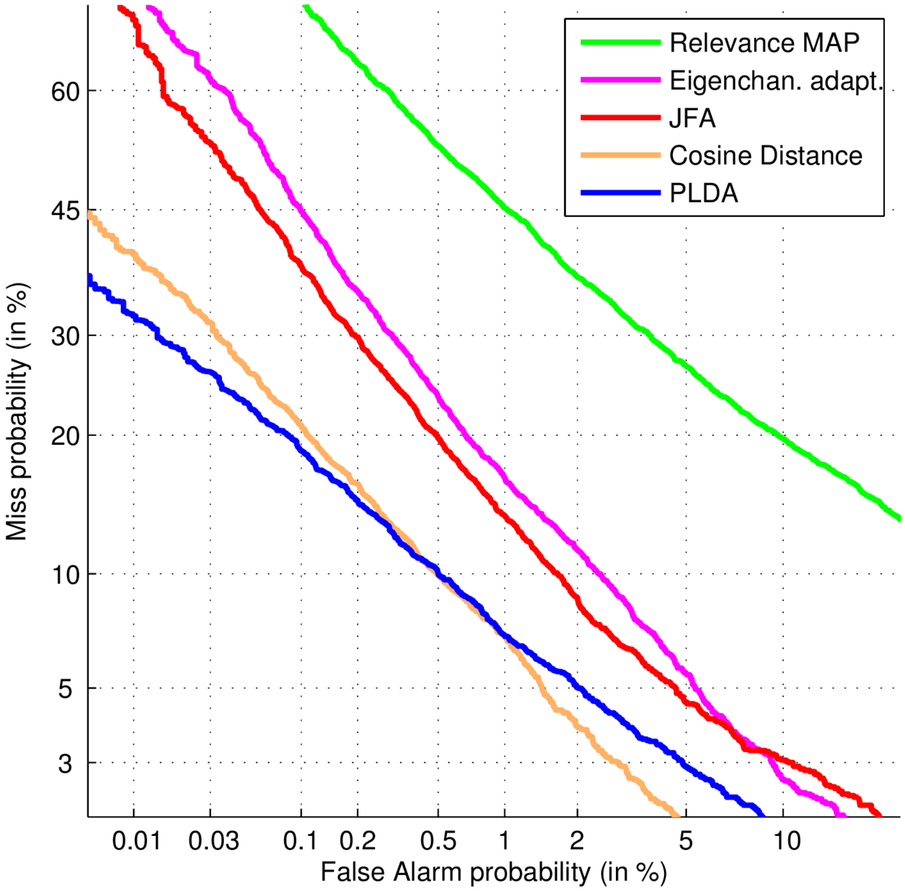


Figure 10.1: Comparison of SRE techniques on female subset of NIST SRE 2010 condition 5

Table 10.1: Comparison of different SRE techniques on a female subset of extended condition 5 (tel-tel) from the NIST SRE 2010 evaluation.

System	DCF$_{\text{new}}$	DCF$_{\text{old}}$	EER
Relevance MAP	0.92	0.54	15.65
Eigenchannel adaptation	0.81	0.25	5.17
JFA	0.75	0.22	4.74
Cosine distance	0.48	0.14	2.88
PLDA	0.41	0.14	3.52

Detailed results showing the favorite metrics are provided in Table 10.1.

10.1.4 Analysis of the Calibration

If we assume, that the final scores provided by the system can be interpreted as the calibrated log-likelihood ratios, then a user of the system is able to analytically select a desired operating point, which best suits his needs and respects the prior probabilities of target and non-target trials present in his data. Here we analyze the actual performance of JFA, cosine distance a and PLDA system over a range of operating points expressed by the effective prior (see Section 3.4). In an ideal situation when system provides perfectly calibrated scores, the actual performance on the evaluation data would be the same as the theoretical best performance expressed by the minDCF metric.

Comparison of the actual performance of individual systems is shown on Figure 10.2. All of the scores of these three systems were calibrated by linear calibration trained on a subset of NIST SRE 2008 data. It can be observed that modeling i–vectors by PLDA has greatly reduced the loss in an actual performance and even if the minDCF is comparable to the cosine distance scoring, the PLDA system has a better practical use as it provides better calibrated scores. When comparing the JFA with the cosine distance scoring of i–vectors, we see similarly problematic calibration. Even though JFA is a generative classifier, its complexity was probably the cause for the bigger difference between the scores obtained on the development set and the scores on an unknown evaluation set. Other comparisons can be seen in the presentation of the ABC system submitted for the NIST SRE 2010 evaluation [Brümmer et al., 2010a].

10.2 Evolution of the PLDA

After the NIST SRE 2010 evaluation, PLDA was in the center of the interest of the research community. Shortly after the NIST workshop and Odyssey 2010 conference in Brno, we have introduced a discriminative way of training the PLDA parameters. It was the BOSARIS workshop in Brno, where both the training using SVM [Cumani et al., 2013] and logistic regression [Burget et al., 2011] were developed.

In Figure 10.3, we can observe the effect of both discriminatively trained PLDA, length normalization and additional condition-dependent mean normalization (mean of the train-

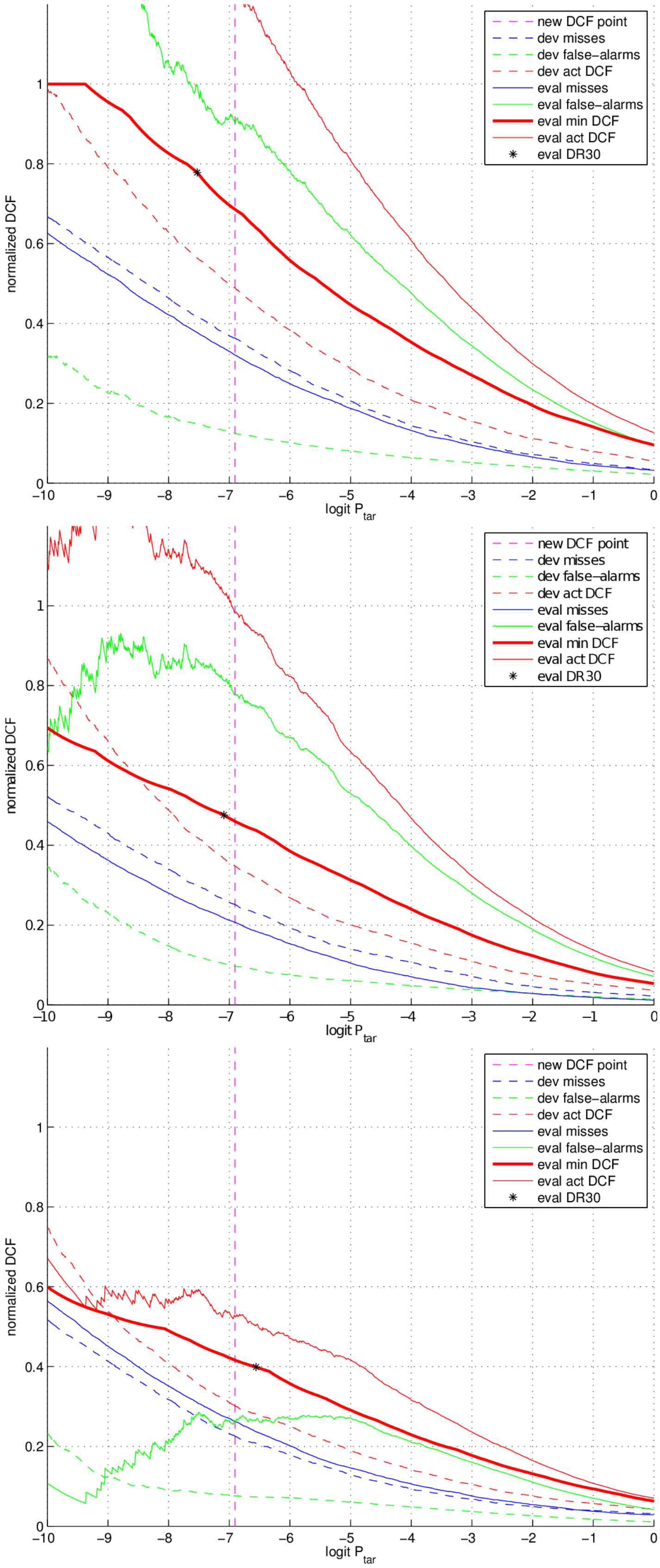


Figure 10.2: Normalized Bayes error-rate plots for three systems evaluated on the female subset of extended condition 5 (tel-tel) of the NIST SRE 2010. The top system is the JFA, in the middle is the i–vector system with cosine distance scoring and in the bottom is the PLDA system without length normalization.

Table 10.2: Comparison of the PLDA variants on extended condition 5 (tel-tel) from the NIST SRE 2010 evaluation.

	Female Subset			Male Subset		
System	$\mathrm{DCF}_{\mathrm{new}}$	$\mathrm{DCF}_{\mathrm{old}}$	EER	$\mathrm{DCF}_{\mathrm{new}}$	$\mathrm{DCF}_{\mathrm{old}}$	EER
PLDA	0.40	0.15	3.57	0.42	0.13	2.86
DPLDA-LR	0.40	0.12	2.94	0.39	0.10	2.22
DPLDA-SVM	0.39	0.11	2.35	0.31	0.08	1.55
PLDA+length norm.	0.35	0.10	2.33	0.32	0.08	1.73
PLDA+length norm.+MR	0.32	0.09	1.98	0.29	0.07	1.30
HT-PLDA	0.34	0.11	2.22	0.33	0.08	1.47

ing i–vectors coming from the telephone data was removed from the evaluation data). All of the PLDA systems are trained on the same dataset as described in the previous section. The baseline PLDA system represented by the blue DET curve is taken from the previous section, the red DET curve represents the discriminatively trained PLDA system, with no length normalization or other transformation of i–vectors. DPLDA was trained with all of the parameters initialized as matrices of zeros. The target prior probability was set to 0.001 to reflect the NIST SRE 2010 primary metric. The regularization was performed by means of early stopping during this experiment. It took approximately 30 iterations for the algorithm to converge.

The Magenta line represents the system with length normalization that was tuned to get the best overall results for all NIST SRE 2010 conditions. In this system, i–vectors were first reduced into 150 dimensions and then the PLDA with both full rank matrices representing speaker and channel subspaces was trained. The last system represented by the black DET curve is a modification of the magenta system which consists only in the condition dependent mean normalization. This has further improved the PLDA system on the telephone condition. It should be noted, that this approach was specific to the particular training list used during these experiments. During our other experiments with the PLDA, we have extended our training list with the additional telephone and microphone data and the positive effect of this condition-dependent mean normalization was reduced.

The discriminative training can apparently deal with the non-Gaussian behavior of the i–vectors and produce significantly better results than the baseline PLDA. However, the discriminative PLDA did not keep the winner's laurel for long time. Shortly after this approach was developed, the length normalization was introduced, and standard PLDA with the i–vector pre-processing, as described in chapter 8, has reached the performance of the heavy-tailed version of the PLDA. It should be noted that the length normalization applied on i–vectors before DPLDA training did not noticeably change the result. Also the DET-curve for the HT-PLDA would practically overlap with the magenta curve representing the standard PLDA with length normalization.

The comparison of all systems on both female and male subset is provided in Table 10.2. We also include an SVM variant of the DPLDA system, which was trained on the same dataset and with same i–vectors as the other systems [Cumani et al., 2011].

An analysis of the calibration loss for the DPLDA system is shown in Figure 10.4. The

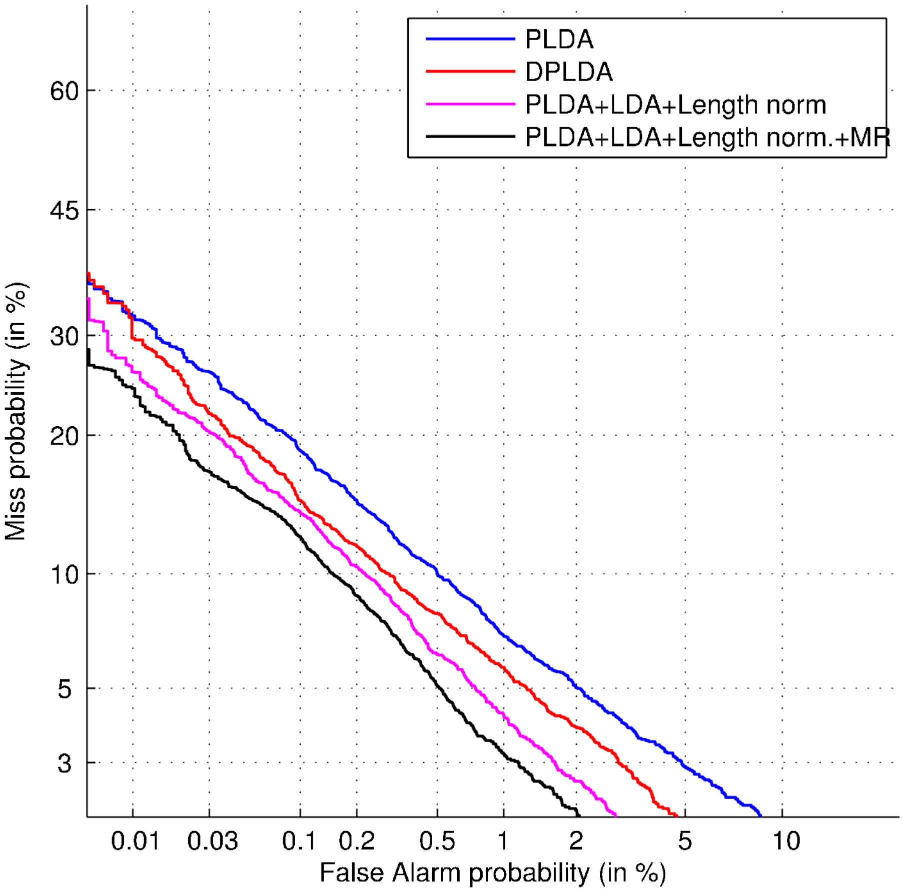


Figure 10.3: Comparison of PLDA systems on female subset of NIST SRE 2010 condition 5: Blue system is a standard PLDA without length normalization, red DET curve represents discriminatively trained PLDA (DPLDA), magenta and black correspond to the standard PLDA system with length normalization and additional condition dependent mean normalization.

scores were calibrated by a linear calibration trained on a subset of NIST SRE 2008 data in the same way in the previously presented systems. In comparison with the standard PLDA and other techniques, the calibration on the DPLDA scores was better, especially around the desired operating point ($\mathrm{DCF}_{\mathrm{new}}$).

10.3 Analysis of DPLDA in Different Acoustic Conditions

After the introduction of the length normalization, we were naturally frustrated by the lower performance of DPLDA versus the standard PLDA. When we were looking at the DET curves of DPLDA evaluated on the training data, it was obvious that the discriminative training can separate almost all training examples. This behavior encouraged us to test the model under various more difficult conditions and with different training data. The ideal opportunity was to use, at that time recently developed, PRISM set (see sec-

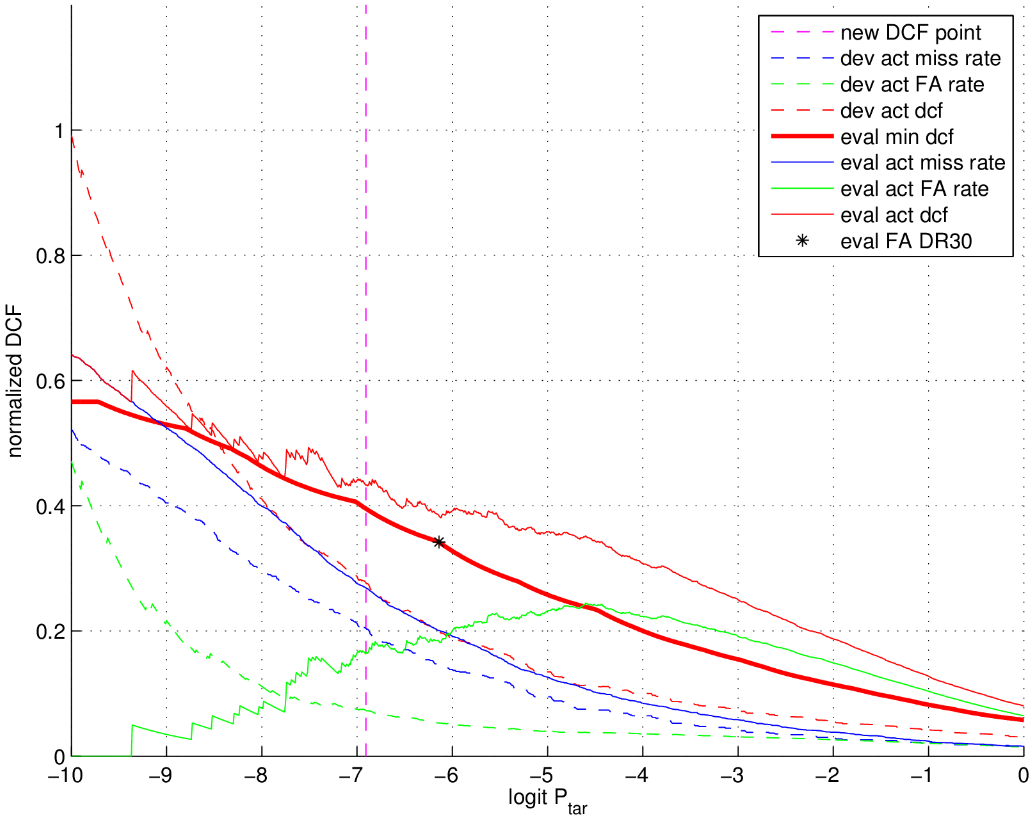


Figure 10.4: Normalized Bayes error-rate plots for DPLDA system evaluated on the female subset of NIST SRE 2010 condition 5. This figure can be compared with Figure 10.2.

tion 2.3) and later on much more difficult data released under the RATS (see section 2.5) program. Although we did not achieve a broad success with the discriminative model, there are some cases where DPLDA can outperform PLDA with length normalization.

10.3.1 Analysis on the PRISM Set

In this series of experiments, the training set, which is common for the UBM, i–vector extractor and PLDA has changed. Similarly to the previous experiments, the training set contains data from Fisher 1 and 2, Switchboard phase 2 and 3 and Switchboard cellphone phases 1 and 2. In addition to the Fisher and Switchboard, also data from the speakers not present in the PRISM evaluation sets and coming from NIST SRE evaluations of the years 2004-2010 are included. We also included segments with added noise and reverberation. The detailed description of the sets is given in [Ferrer et al., 2011b]. To represent different types of channel variability, we chose the same PRISM subsets as in [Ferrer et al., 2012] and briefly described in section 2.3.

Apart from the change in the datasets, we used a gender-dependent 2048-component diagonal covariance UBM and the dimensionality of i–vectors was raised to 600. With the increase of i–vector dimensionality, we reduced also the LDA compression to 200 dimensions prior to the baseline Gaussian PLDA training. In this case, we did not use any condition-dependent mean removal. The feature extraction and voice activity detection remained the same as described in section 10.1.1.

The DPLDA system parameters were always initialized from zeros and an initial configuration was inspired by the baseline PLDA system with the dimensionality of both speaker and channel subspaces set to 200. We will show a set of representative experiments conducted on various channel conditions during our search for an ideal DPLDA configuration.

Tuning the Regularization

The first experiments performed on the PRISM set were devoted to tuning the regularization parameter, which we did not address in the previous DPLDA system. We took the initial configuration of DPLDA and swept the values of regularization parameter.

We present the results of the sweep in various conditions in figures 10.5 and 10.6. The bold blue DET curve represents our baseline Gaussian PLDA system (marked as GPLDA), while the others represent different DPLDA systems trained with various values of the regularization constant. The dashed light-blue lines represent boundaries of the “Doddington’s rule of 30” (see Section 3.2) computed from the baseline PLDA. For each system, we compute the $\mathrm{DCF}_{\mathrm{new}}$ value and represent it by a color point.

Tuning the regularization parameter is important, as we can see from figures 10.5 and 10.6, that too little or too much regularization leads to poor results. After the sweep, we finally chose the regularization value of 0.1 for our next experiments. We can observe that with the exception of the “lan” condition (second graph in figure 10.5), where trials are formed by two telephone conversations of the same language (English, Chinese, Russian, Arabic or Thai), the baseline is always better in the low-false-alarm regions and therefore also on the $\mathrm{DCF}_{\mathrm{new}}$ metric. In some other conditions (“tela” and “noi”), the DPLDA can match or slightly outperform the baseline in the region of higher false alarm rates. In the reverberated data (“rev” condition), the DPLDA did not outperform our baseline at any operating point.

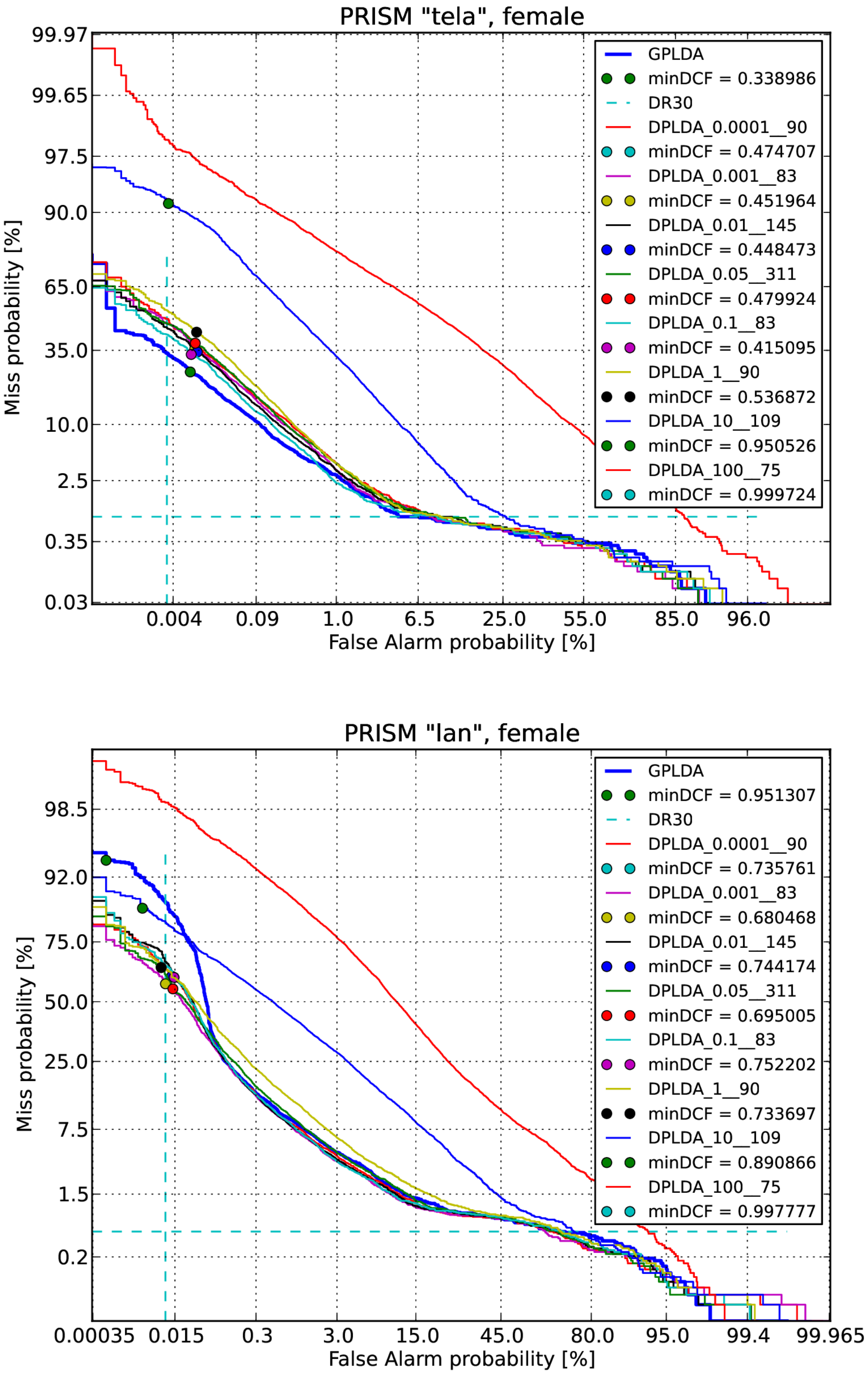


Figure 10.5: Tuning of the regularization for the DPLDA: Blue system represents the generative PLDA baseline. The X and Y in format DPLDA_X__Y in the legends correspond to the regularization constant and number of training iterations, respectively.

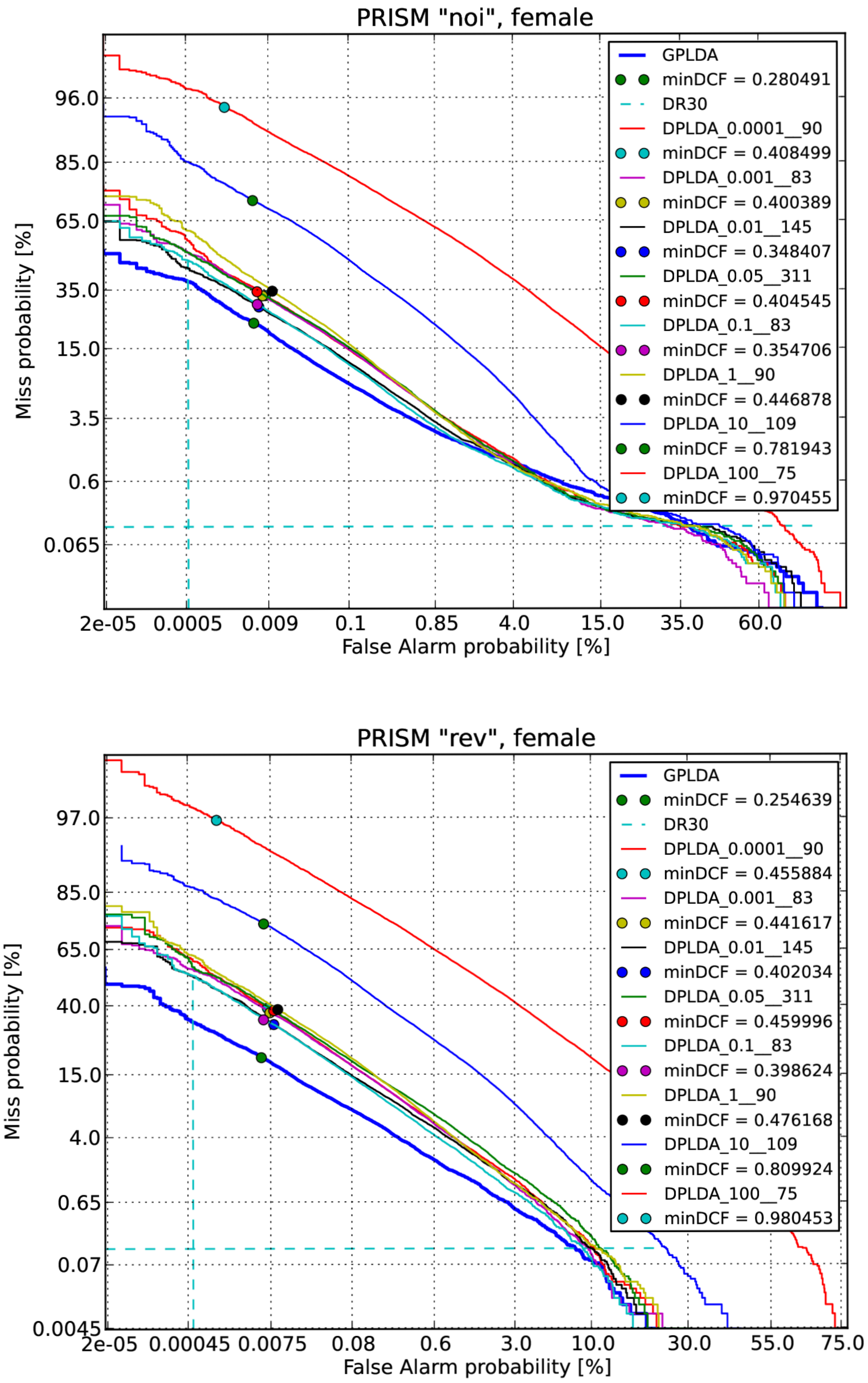
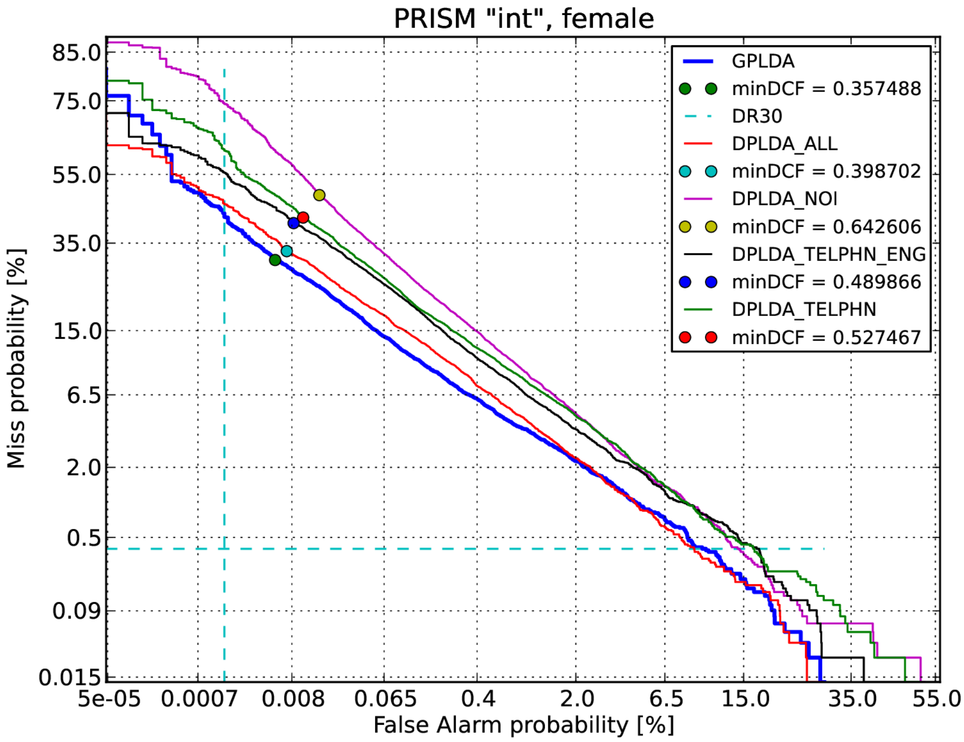


Figure 10.6: Tuning of the regularization for the DPLDA: Blue system represents the generative PLDA baseline. The X and Y in format DPLDA_X__Y in the legends correspond to the regularization constant and number of training iterations, respectively.

Training with Matched Datasets

Having the test conditions conveniently divided according to different nuisance attributes, we also tested the DPLDA in the matched-training scenario. We divided our training according to three criteria: “NOI” – 4035 segments representing only the segments with added noise, “TELPHN_ENG” – 24300 of only clean English telephone segments and “TELPHN” for all 39604 clean telephone segments. We wanted to test whether the algorithm can find a better solution for a hard condition (noisy data), by training only on such data and not being overwhelmed by the vast majority of trials formed from the easier clean data. Also we wanted to test the scenario, where we minimize the variability and train on the clean data very closely representing the test data. This would be training on only English clean telephone data and testing on a corresponding condition. We also included training on all telephone data which brings more variability, but increases the amount of training data.

We present the results of selected experiments for different conditions and different training sets in figures 10.7 and 10.8. It is clear that using all available data for training (56348 segments) gives the best results. Even in the scenarios with relatively large training set on English telephone data and English telephone tests (Figure 10.8, top graph), we were not able to achieve better performance than with the system trained with all training data. The trend is always in favor of more data regardless the target test. These findings lead us to the intuition that the discriminative training of PLDA needs even more, and at the same time harder, training data in order to reasonably generalize and compete with the standard PLDA.



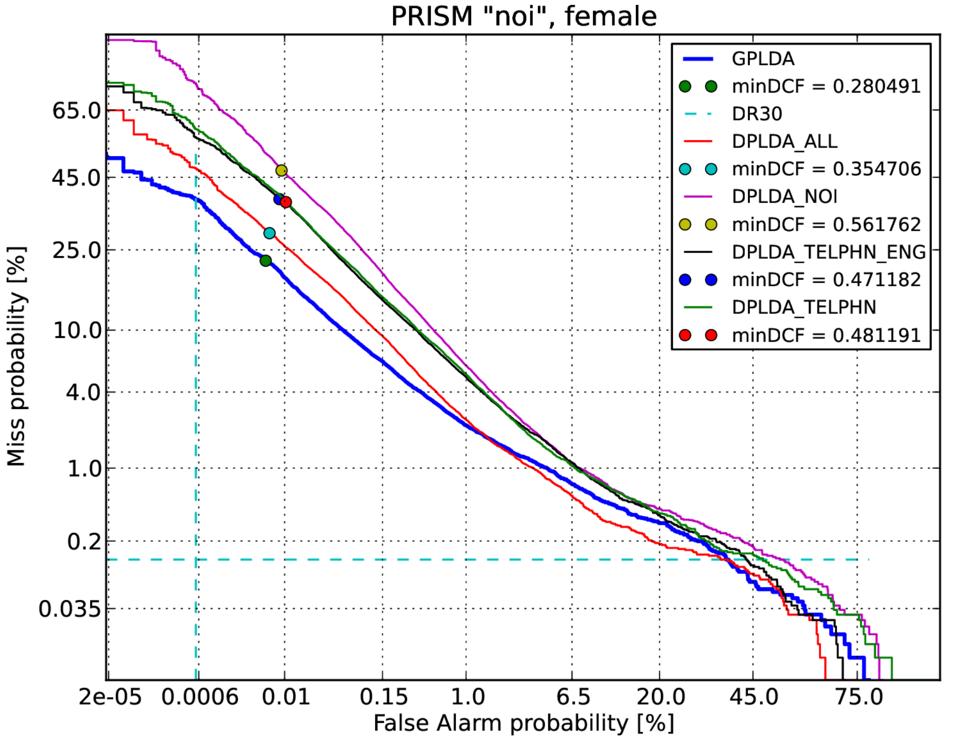


Figure 10.7: Analysis of DPLDA when training on matched data and other training sets. Top graph corresponds to testing on English interviews recorded over different microphones. Bottom graph corresponds to testing on data with added noise.

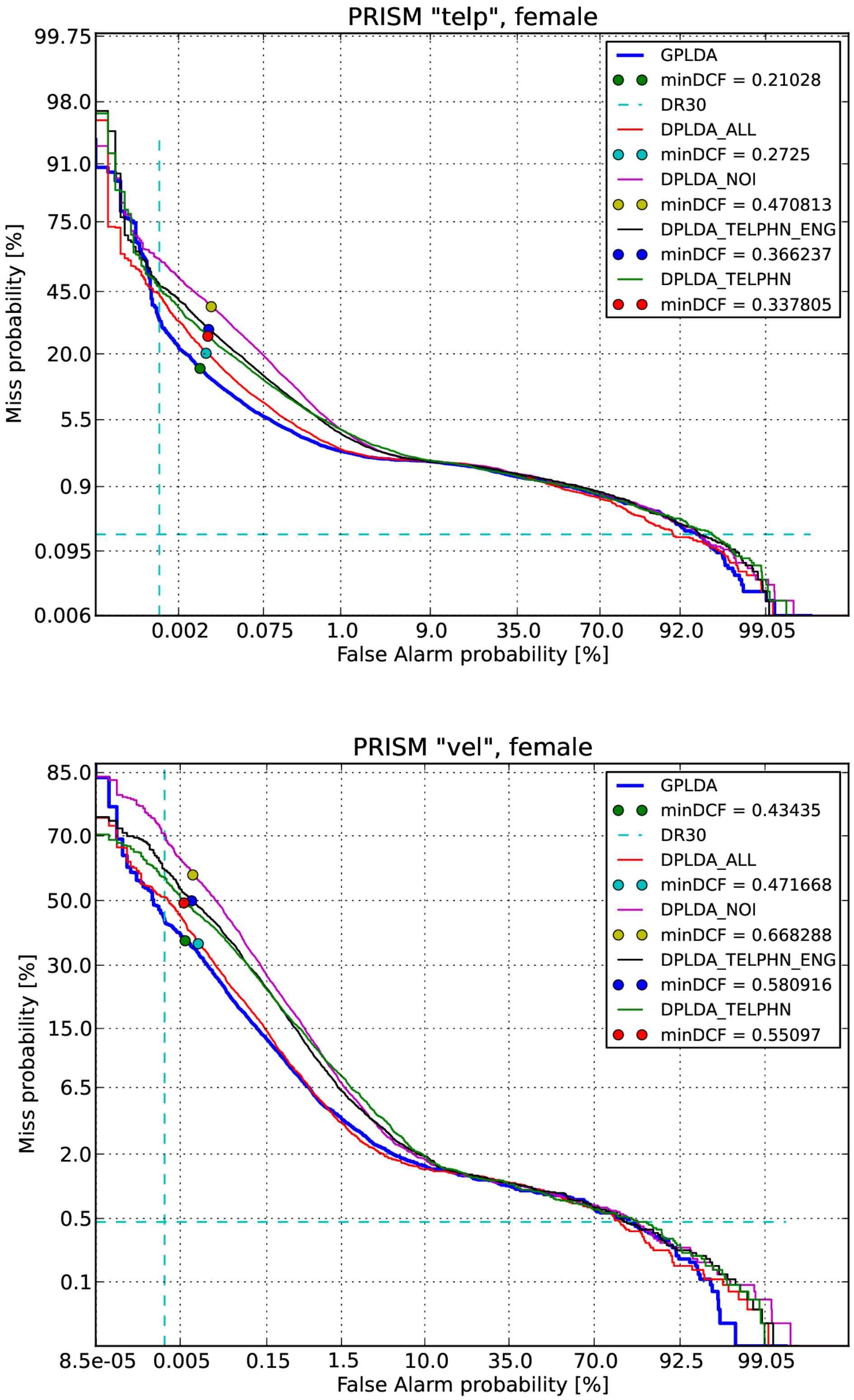


Figure 10.8: Analysis of DPLDA when training on matched data and other training sets. Top graph corresponds to testing on English telephones. Bottom graph corresponds to testing on trials formed from normal vocal effort English conversations in enroll and various vocal efforts in test.

10.3.2 Analysis on RATS Data

Evaluating SRE performance on the RATS data poses many more challenges than simply taking the state-of-the-art system and running it on the data. This extremely noisy data has brought a lot of attention to developing different variants of robust acoustic features and voice activity detection. It would be out of the scope of this work to discuss the RATS-specific techniques and we refer the reader to a general system description [Plchot et al., 2013] of our submission for the RATS evaluation in 2013, from which we derive our baseline system.

Our systems again use the same configuration of MFCC features with the exception of using 25 ms analysis window versus 20 ms. We train a diagonal, 2048-component, gender-independent UBM and extract 600-dimensional i–vectors. It should be noted, that the baseline system we use here is not exactly the same as listed in the system description. The systems presented there use much larger set for training the PLDA system, which would make our experiments with DPLDA very time-consuming and expensive. All of our systems submitted for the RATS evaluation were duration-independent, which will be also the baseline scheme for our experiments here.

It is important to mention the composition of the training set for PLDA. After tuning the composition of our training data, the general consensus was to use as many short cuts from the segments as possible along with the original long segments. The reason for this composition is greatly influenced by the evaluations, where the emphasis is put on the performance obtained on the 30 s and 10 s cuts. There is also a 3 s and 120 s test condition in RATS SRE evaluation protocol. The 120 s condition is getting less attention as the program goals for this test were mostly achieved. The 3 s condition was considered too hard especially in the first two phases of the RATS project and we did not focus on tuning for these durations.

The final training list for our baseline PLDA system was a compromise between the performance on the short duration segments and a reasonable amount of data for training the DPLDA system. In total, it contained 210 thousand segments, out of which 70 thousand were randomly selected 30 s cuts and another 70 thousand were randomly selected 10s cuts. The training of PLDA followed the same recipe as previously described, with LDA dimensionality reduction to 200 dimensions and length normalization. Corresponding DPLDA systems were trained using parameters initialized to zeros. As the trials in the RATS SRE evaluations are defined as multi-session (6 enrollment segments versus one test), our development test sets also follow this scheme. In order to obtain the scores with the DPLDA system, we used i–vector averaging to represent the multi-session trial as a standard one-to-one i–vector trial. We performed the multi-session scoring with standard PLDA, but it should be noted that doing the averaging does not significantly change the results.

Results of the experiments reported on the metrics of the RATS program are summarized in table 10.3. We report only results obtained on the RATS Patrol team development test sets as the key for the official evaluation set of the program was not available at the time of writing this text.

It can be seen that training both systems on the whole dataset yields slightly worse performance than training a duration-dependent system. Also the DPLDA system is

Table 10.3: Comparison of the PLDA and DPLDA systems trained on all data, 10s segments or 30 s segmets. Results are given on the RATS Patrol development sets. 30s-30 s and 10s-10 s correspond to the duration of the enrollment and test utterances. The metrics are FA_10, which correspond to the false alarm rate at miss rate 10% and MISS_2.5 is a miss rate at false alarm rate 2.5%. EER stands for equal-error rate.

	30 s – 30 s			10 s – 10 s		
System	FA_10	MISS_2.5	EER	FA_10	MISS_2.5	EER
PLDA all	3.53	13.36	6.21	10.04	27.01	10.04
DPLDA all	3.68	13.89	6.30	10.03	28.11	10.02
PLDA 30 s	3.32	12.62	6.06	9.99	26.41	9.99
DPLDA 30 s	**3.12**	**12.09**	**5.81**	9.29	26.43	9.66
PLDA 10 s	3.54	13.21	6.17	9.29	**25.75**	9.65
DPLDA 10 s	3.48	13.24	6.08	**9.01**	25.94	**9.49**

performing slightly worse than the PLDA system when trained on all data. The situation has finally turned in favor of DPLDA when training duration-dependent systems. In these scenarios, the DPLDA outperformed PLDA on almost all metrics.

So much as we can be pleased by finally obtaining results that could beat the PLDA with length normalization, it is fair to note that with classical PLDA, we are able to use much larger datasets for training and build a big, well performing and condition independent system. In fact, the best systems in our submission for RATS 2014 evaluations were using more than 2.5 million segments of various lengths for training. Even with our parallel implementation of DPLDA training, it is unfeasible to experiment with such systems.

10.4 Full Posterior Distributions PLDA

In case of short speech segments, the covariance of the i–vector posterior distribution is large (i.e. the i–vector estimated as a MAP point estimate does not sufficiently approximate its posterior distribution). The proposed Full Posterior Distribution PLDA model address this problem by integrating over all possible realizations of i–vectors generated from its posterior distribution. We will show that this approach is superior to the standard PLDA in case of short segments (less than 60 seconds). In case of long segments, the i–vector obtained as a MAP point estimate is already a good representation of the i–vector posterior distribution and we will show that in such cases the newly proposed model is equivalent to the original PLDA.

As the focus of FPD-PLDA is mainly on short utterances, we defined a dataset that consists of speech segments, from NIST SRE10 extended core condition, which were cut, after Voice Activity Detection, to obtain segments of variable duration in the range 3–30, 10–30, 3-60, and 10–60 seconds, respectively. These sets of segments have been scored according to the official NIST SRE 2010 conditions 1–5 [NIST, 2010], which are summarized in Table 2.2.

We used a similar configuration of cepstral features as in the previous experimiments.

Using a 25 ms Hamming window with the shift of 10ms, we extract 19 MFCC coefficients together with log-energy. These 20-dimensional feature vectors were subjected to short time mean and variance normalization using a 3 s sliding window. Delta and double delta coefficients were then computed using a 5-frame window giving 60-dimensional feature vectors. Voice activity detection was performed in the same way as described in Section 10.1.1.

The i–vector extractor was based on a 2048–component full-covariance gender-independent UBM, trained using NIST SRE 2004–2006 data. Gender-dependent i–vector extractors for the baseline system (marked as "std" in the tables) were trained using the data of NIST SRE 2004–2006, Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, Fisher English Parts 1 and 2.

All experiments were performed using i–vector posteriors with dimensionality 400. The PLDA was trained with a speaker variability sub–space of dimensionality 120, and full channel variability sub–space. Although both female and male speaker tests were performed, we report more detailed results on the female datasets only, because the NIST SRE 2010 core test on female speakers is known to be more difficult, thus more often compared in the literature. The results on the male speakers confirm the ones reported for female speakers, as will be shown in 10.7. It should be also noted that for all experiments with FPD-PLDA on NIST SRE 2010, the standard approach to scoring was used (see sections 7.5.3 and 7.5.4). The effect of asymmetric FPD-PLDA will be demonstrated later when testing on NIST SRE 2012.

Table 10.4 summarizes the results of the tests performed on the NIST SRE 2010 female extended conditions, including the core condition (condition 5), in terms of percent Equal Error Rate and normalized minimum Detection Cost Function (DCF_{old} and DCF_{new}) as defined by NIST for SRE08 and SRE10 evaluations [NIST, 2010]. In this table, the PLDA and FPD–PLDA systems are compared using the original interview data, or telephone conversations, without any cut. Labels "tel" and "tel+mic" refer to the datasets used for training the PLDA parameters, including telephone data only, or additional microphone data. Labels "Std" and "FPD" refer to the standard and the Full Posterior Distribution PLDA, respectively. The first row rows gives the baseline results, obtained using standard PLDA trained on telephone data only. Second row shows a situation when standard PLDA is used for training the model parameters and the FPD-PLDA is used for scoring. In the third row, the FPD-PLDA was used both for training and scoring. The last three rows show the effect of adding microphone data in training the PLDA parameters: sensible performance improvement with respect to telephone-only list is obtained, excluding, as expected, the tel–tel condition 5.

Results are given for the five NIST 2010 conditions. It can be observed that the matched conditions 5 and 1 — tel–tel and int–int, respectively, achieve the best results, whereas the difficulty of the task decreases from condition 2 to condition 4. The same trend is confirmed for all experimental conditions, shown in the table 10.4, and later it will be also the case for the other tests using variable duration segments.

The new model not only keeps the accuracy of the standard model for long segments, but also shows a small improvement for EER and DCF_{old} in three conditions (2,3,4). The third and last row present the effect of using the i–vector covariance also in training. Since the training segments have long durations and corresponding i–vectors are already good estimates of the i–vector posterior distribution, the results are similar to the ones reported in the second fifth row where the standard PLDA is used for training.

Since the systems trained with the “tel” list perform worse than those trained with the “tel+mic” list, all the remaining experiments on the NIST 2010 data, have been performed with the latter. In its first three rows, Table 10.5 compares the performance of the PLDA and FPD–PLDA classifiers using the two length–normalization methods described in Chapter 8 on the 3–60 seconds cuts. The results of the last row show that again, there is no advantage in using the full i–vector posterior in training the PLDA models. The effect of the two length–normalization approaches is comparable, thus in the following we will present only the results obtained with the Projected Length Normalization (FPD2) (8.8).

The tests on variable duration cuts, randomly chosen from the extended NIST SRE2010 female set, are shown in Table 10.6. FPD–PLDA shows always a relative improvement, quite small for long enough segments, but up to 20% depending on the average duration of the small cuts. The results given in Table 10.7 confirm same trends in performance for male speakers.

Table 10.4: Results for the core extended NIST SRE2010 female tests in terms of % EER, normalized minDCF$_{\text{old}}$ ×1000 and normalized minDCF$_{\text{new}}$ ×1000 using different training lists and PLDA models. Label "tel" and "tel+mic" refer to the datasets used for training the PLDA, including or not microphone data. "Std" and "FPD" labels refer to standard PLDA and FPD–PLDA, respectively. I–vector posterior length–normalization is performed by means of (8.8).

List	Train	Test	condition 2			condition 3			condition 4			condition 1			condition 5		
			EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$
tel	Std	Std	4.2	224	641	2.5	113	445	1.7	102	411	2.0	84	346	2.0	100	339
tel	Std	FPD	3.9	214	638	2.3	111	462	1.6	101	419	1.7	81	346	2.0	100	346
tel	FPD	FPD	3.9	214	635	2.4	110	450	1.6	99	415	1.8	79	345	2.0	98	336
tel+mic	Std	Std	2.6	124	460	2.2	103	405	1.1	65	303	1.8	68	258	1.9	105	335
tel+mic	Std	FPD	2.3	114	455	2.1	103	402	1.0	60	296	1.7	63	254	2.0	103	347
tel+mic	FPD	FPD	2.3	112	455	2.0	100	396	1.0	59	288	1.6	60	253	2.0	101	344

Table 10.5: Results for cuts of 3–60 second test data, using different length–normalization approaches. The PLDA parameters are trained using both microphone and telephone data. Labels "Std" and "FPD" refer to standard PLDA and FPD–PLDA, respectively, and the numeric suffix of FPD corresponds to the i–vector posterior length–normalization method.

Train	Test	condition 2			condition 3			condition 4			condition 1			condition 5		
		EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$
Std	Std	9.1	384	812	7.8	368	832	7.3	312	695	7.0	273	630	6.7	337	729
Std	FPD1 (eq. 8.7)	6.7	327	791	6.1	343	838	5.2	259	676	4.8	232	603	6.2	322	722
Std	FPD2 (eq. 8.8)	6.7	328	791	6.2	343	838	5.2	259	676	4.7	232	603	6.2	323	722
FPD2	FPD2	6.5	327	796	6.3	355	837	5.0	255	676	4.6	229	601	6.3	328	731

Table 10.6: Results for cuts of variable duration test data, randomly chosen from the extended NIST SRE2010 female tests, in terms of % EER, normalized minDCF$_{\text{old}}$ × 1000 and normalized minDCF$_{\text{new}}$ × 1000 using different PLDA models. The PLDA parameters are trained using both microphone and telephone data, labels “Std” and “FPD” refer to standard PLDA and FPD–PLDA, respectively. I–vector posterior length–normalization is performed by means of (8.8).

Test	Duration	condition 2			condition 3			condition 4			condition 1			condition 5		
		EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$	EER	DCF$_{\text{old}}$	DCF$_{\text{new}}$
Std	3–30	12.4	531	921	11.3	521	915	11.1	441	864	9.8	405	794	10.6	493	915
FPD	3–30	9.8	474	901	9.3	498	929	8.3	382	849	7.6	327	756	9.7	475	912
Std	10–30	9.0	431	890	8.6	429	900	6.6	318	820	7.0	317	707	7.6	390	856
FPD	10–30	7.7	388	873	7.5	417	893	5.7	285	785	5.5	278	650	7.2	373	836
Std	3–60	9.1	384	812	7.8	368	832	7.3	312	695	7.0	273	630	6.7	337	729
FPD	3–60	6.7	328	791	6.2	343	838	5.2	259	676	4.7	232	603	6.2	323	722
Std	10–60	7.0	318	787	5.0	283	777	4.7	227	636	4.9	211	558	4.9	265	701
FPD	10–60	5.7	283	761	4.8	271	806	3.9	200	603	4.1	176	555	4.7	260	693
Std	Full	2.6	124	460	2.2	103	405	1.1	65	303	1.8	68	258	1.9	105	335
FPD	Full	2.3	114	455	2.1	103	402	1.0	60	296	1.7	63	254	2.0	103	347

Table 10.7: Results for cuts of variable duration test data, randomly chosen from the extended NIST SRE2010 male tests, See Table 10.6 captions.

Test	Duration	condition 2			condition 3			condition 4			condition 1			condition 5		
		EER	DCF_{old}	DCF_{new}	EER	DCF_{old}	DCF_{new}	EER	DCF_{old}	DCF_{new}	EER	DCF_{old}	DCF_{new}	EER	DCF_{old}	DCF_{new}
Std	3–30	8.3	379	825	9.8	448	923	8.9	364	766	6.0	280	697	9.4	436	857
FPD	3–30	6.2	325	795	8.0	432	929	6.6	308	747	4.3	224	641	8.6	419	842
Std	10–30	5.7	286	777	6.8	368	892	5.8	273	701	4.2	192	607	6.7	326	811
FPD	10–30	4.7	243	741	6.1	326	877	5.1	240	665	3.1	157	529	6.3	308	771
Std	3–60	5.8	259	645	6.3	284	753	5.9	247	596	4.5	182	464	6.3	286	692
FPD	3–60	4.1	204	605	5.6	276	819	4.1	194	540	3.0	136	402	5.3	269	697
Std	10–60	3.8	196	609	5.1	251	738	3.5	172	547	2.5	116	402	4.6	224	627
FPD	10–60	2.9	159	565	4.5	231	744	3.0	149	523	2.0	88	370	4.2	218	631
Std	Full	1.1	57	270	1.9	86	353	1.2	47	200	0.6	28	138	1.5	82	310
FPD	Full	0.9	47	249	1.7	83	356	1.1	45	192	0.5	24	121	1.4	84	319

10.5 Comparison on NIST 2012

Pooled results for female and male speakers are reported in Table 10.8 for the NIST 2012 SRE evaluation experiments described below. In these experiments, the acoustic features were again 60–dimensional MFCCs, modeled with a 2048 components full–covariance UBM. The i–vector dimension was increased to 600. Moreover, Linear Discriminant Analysis was performed to reduce the i–vector dimensionality to 200, before applying i–vector whitening and length normalization. Since the resulting i–vectors are already small, no dimensionality reduction was applied for the speaker subspace, i.e. the speaker subspace in PLDA was set to 200. The UBM was trained on speech segments taken from the NIST 2004, 2005, 2006, 2008 and 2010 evaluation corpora, and from the enrollment set of NIST 2012 evaluation. Additionally, the Fisher, Switchboard Phase 2 and Switchboard Cellular datasets were used to train the i–vector extractor. For training the PLDA, only Switchboard Phase 2 and Switchboard Cellular datasets were were added to the NIST datasets. Due to the enormous amount of trials involved in the evaluation (some tens of millions), we did not test the complete FPD–PLDA approach. Since NIST 2012 enrollment segments are on average quite long, we were able to test FPD–PLDA according to the Asymmetric FPD–PLDA approach described in Section 7.5.4. Moreover, we had empirical evidence that representing a target speaker by means of a single i–vector, computed as the average of all its i–vectors, provides higher accuracy with respect to the standard multi–session PLDA scoring. The same approach was, thus, followed for obtaining the FPD–PLDA scores.

The results comparing standard PLDA and Asymmetric FPD–PLDA are given in Table 10.8 in terms of minimum and actual $C_{primary}$. Note, that in contrast to minDCF, there is no analytic version of the “minimum” $C_{primary}$. By “minimum”, we mean a $C_{primary}$ as defined by NIST, but with calibration performed on the evaluation data, while the “actual” denotes a correct calibration trained on the development set of scores. The development scores were formed out of NIST SRE 2004–2010 data, which were truncated to the expected duration (NIST has released the information about the average duration of test segments before evaluation). Also, the crowd and the HVAC noise was added to the portion of this data.

These results show that the Asymmetric FPD–PLDA is almost equivalent to the standard PLDA. For minimum $C_{primary}$, there is an improvement for conditions 2 and 5, which include short and variable duration segments. An excellent result have been obtained with discriminatively trained PLDA in terns of the actual $C_{primary}$, where the calibration loss for DPLDA system is low compared to the other two techniques. This can can indicate that the scores from DPLDA are more “robust” (even though less discriminable for conditions 2, 4, 5) in terms of being good log-likelihood ratios than scores obtained from generative PLDA. These results confirm that DPLDA is a technique with a built-in calibration, which is a very useful property for a real use scenario.

Table 10.8: NIST SRE 2012 core-extended test: comparison of DPLDA, PLDA and Asymmetric FPD–PLDA on minimum and actual $C_{primary}$. The numbers associated to the conditions refer to the mean duration of the segments, after voice activity detection, and to the corresponding standard deviation.

System	Condition 1 interview without added noise 45s – 41	Condition 2 phone call without added noise 56s – 48	Condition 3 interview with added noise 75s – 37	Condition 4 phone call phone with added noise 110s – 56	Condition 5 phone call from a noisy environment 57s – 48
DPLDA (min)	0.230	0.261	0.206	0.287	0.249
PLDA (min)	0.255	0.206	0.244	0.265	0.222
FPD–PLDA (min)	0.253	0.193	0.241	0.264	0.211
DPLDA (act)	0.250	0.300	0.215	0.339	0.333
PLDA (act)	0.336	0.292	0.294	0.370	0.342
FPD–PLDA (act)	0.336	0.292	0.293	0.389	0.344

Chapter 11

Conclusions

This work proposes two variants of the Probabilistic Discriminant Analysis, which, in its standard form, is currently considered as the state-of-the art technique in the text-independent speaker recognition. Preceding state-of-the art techniques have been put into the context with the standard PLDA, which also serves as a baseline for the proposed modifications. The performed comparison of all techniques on the NIST SRE 2010 dataset presents a historical progress in the SRE technology. In Figure 10.1, we can identify two milestones in the SRE technology. It is an introduction of the channel compensation techniques and using i–vectors as low-dimensional, information-rich features for modeling.

Discriminative PLDA

The functional form of the standard PLDA model for evaluating the speaker verification trial has been used as the basis for designing the discriminative approach to training of PLDA parameters. A single discriminative model then directly addresses the symmetric speaker verification task: a discrimination between the same- and different-speaker trial formed by two i–vectors. Although the discriminative training was initially bringing substantial improvements with respect to the original PLDA, after the application of the length normalization of i–vectors, the standard PLDA model achieves slightly better performance in the minimum DCF and EER metrics.

The performed comparative study of PLDA and DPLDA in various acoustic environments has also shown slightly better overall performance of the standard generative PLDA in terms of minimum DCF and EER evaluation metrics. In the domain of highly degraded RATS data, the discriminative approach has shown small improvements in the duration-dependent systems with respect to generative baseline. These experiments, however, show a theoretical best possible performance not taking into account any calibration loss.

Minimizing the cross-entropy error function as an objective for discriminative training of DPLDA forces the system to output scores in form of calibrated log-likelihood ratios for the wide range of operating points. The possibility of weighting individual trials allows for focusing on the area around the desired operating point of the system already during training, which makes the consecutive calibration step less necessary. The quality of the calibration of the DPLDA scores has been confirmed by the experiments where the calibration loss on an unseen evaluation set is lower than for the other PLDA variants. Note that all systems are calibrated using separate development set. The low calibration loss for the DPLDA suggests that its scores are not that data-set specific and it is not so necessary to calibrate for each evaluation data-set. This behavior is a desirable property in a real use scenario, where the actual error rates matter much more than the theoretical minimum error rates.

Full Posterior Distribution PLDA

A generative PLDA model that exploits the uncertainty of the i-vector extraction process has been presented. The basic principle is the formulation of the PLDA likelihood, which has been derived for a Gaussian PLDA model based on the i–vector posterior distribution. The new formulation of likelihood evaluation defines a new PLDA model, where the intra–speaker variability is assumed to have a segment–dependent distribution.

Taking into account the posterior distribution of all i–vectors representing an utterace also leads to the need to normalizing this distribution in line with the already established length normalization of i–vectors. Two i–vector pre–processing techniques complying with the new PLDA model have been proposed and their effects were compared in terms of system accuracy. It was shown that an approximate version of a linearized length normalization is sufficiently accurate.

The complexity of the PLDA and FPD–PLDA implementations has been analyzed and an Asymmetric FPD–PLDA approach has been proposed. The asymmetric approach allows for a substantial complexity reduction in a practical detection scenario with known target speakers.

The results obtained both on the extended core tests and on short cuts of different duration of the NIST SRE 2010, and on the extended tests of NIST SRE 2012, confirm that the FPD–PLDA outperforms PLDA mostly for short test segments with variable duration. No loss in the performance has been observed for the standard tests containing long test segments. It was also experimentally demonstrated that for the scenarios when sufficiently long utterances are available for training the PLDA model, we can use the standard PLDA for training and FPD-PLDA for scoring. Therefore in most real use cases, there is no need to perform more expensive FPD-PLDA training.

Future Work

FPD-PLDA can clearly outperform the baseline when testing on short utterances and DPLDA excels at producing well-calibrated scores. Therefore both techniques present a viable option for a real use and should be evaluated in production systems. In my opinion, there are more unknowns in the discriminative approach to be explored. A possible direction for future research could be to address the problem of overtraining the model on the training data and propose more sophisticated ways of regularization. Also an automatic forming of all possible trials in the discriminative training by taking all possible i–vector pairs does not correspond to the real test and could be redesigned. For example, forming the trials out of the same utterance, just recorded over different microphone introduces many artificial positive examples, should be avoided. From the perspective of the functional form for scoring, other blocks can be added to simulate the i–vector pre-processing or condition-dependent calibration.

Bibliography

[Adami et al., 2003] Adami, A., Mihaescu, R., Reynolds, D., and Godfrey, J. (2003). Modeling prosodic dynamics for speaker recognition. In *Proc. of ICASSP, Hong Kong, China*, pages 788–791.

[Auckenthaler et al., 2000] Auckenthaler, R., Carey, M., and Lloyd-Thomas, H. (2000). Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, 10(1-3):42–54.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Boll, 1979] Boll, S. (1979). Suppression of acoustic noise in speech using spectral subtraction. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 27(2):113 – 120.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA. ACM.

[Brümmer, 2004] Brümmer, N. (2004). Spescom DataVoice NIST 2004 system description. In *Proc. NIST Speaker Recognition Evaluation 2004*, Toledo, Spain.

[Brümmer, 2006] Brümmer, N. (2006). A farewell to SVM: Bayes factor speaker detection in supervector space. http://sites.google.com/site/nikobrummer/.

[Brümmer, 2009] Brümmer, N. (2009). The EM algorithm and minimum divergence: Technical report, Agnitio Research, South Africa. `https://sites.google.com/site/nikobrummer/EMandMINDIV.pdf`.

[Brümmer, 2009] Brümmer, N. (2009). EM for JFA: Technical report, Agnitio Research, South Africa. `https://sites.google.com/site/nikobrummer/EMforJFA.pdf`.

[Brümmer, 2010a] Brümmer, N. (2010a). EM for PLDA: Technical report, Agnitio Research, South Africa. `https://sites.google.com/site/nikobrummer/EMforPLDA.pdf`.

[Brümmer, 2010b] Brümmer, N. (2010b). *Measuring, Refining and Calibrating Speaker and Language Information Extracted from Speech.* PhD thesis, University of Stellenbosch.

[Brümmer et al., 2010a] Brümmer, N., Burget, L., Kenny, P., Matějka, P., de Villiers, E., Karafiát, M., Kockmann, M., Glembek, O., Plchot, O., Baum, D., and Senoussauoi, M. (2010a). ABC presentation for NIST SRE 2010. `http://www.fit.vutbr.cz/research/groups/speech/publi/2010/ABC%20System%20description_NIST%20SRE%202010_slides.pdf`.

[Brümmer et al., 2010b] Brümmer, N., Burget, L., Kenny, P., Matějka, P., de Villiers, E., Karafiát, M., Kockmann, M., Glembek, O., Plchot, O., Baum, D., and Senoussauoi, M. (2010b). ABC system description for NIST SRE 2010. In *Proc. of NIST 2010 Speaker Recognition Evaluation, Brno, Czech Republic*, pages 1–20.

[Brümmer and de Villiers, 2010] Brümmer, N. and de Villiers, E. (2010). The BOSARIS toolkit. http://sites.google.com/site/bosaristoolkit/.

[Brümmer and de Villiers, 2010] Brümmer, N. and de Villiers, E. (2010). The speaker partitioning problem. In *Proc. of Odyssey 2010*, Brno, CZ.

[Brümmer and du Preez, 2006] Brümmer, N. and du Preez, J. (2006). Application-independent evaluation of speaker detection. *Computer Speech & Language*, 20(2-3):230–275.

[Burget, 2004] Burget, L. (2004). *Complementarity of Speech Recognition Systems and System Combination*. PhD thesis, Brno University of Technology.

[Burget et al., 2008] Burget, L., Brummer, N., Reynolds, D., Kenny, P., Pelecanos, J., Vogt, R., Castaldo, F., Dehak, N., Dehak, R., Glembek, O., Karam, Z., Noecker, J. J., Na, Y. H., Costin, C. C., Hubeika, V., Kajarekar, S., Scheffer, N., and Černocký, J. (2008). Robust speaker recognition over varying channels. Technical report, Johns Hopkins University.

[Burget et al., 2007] Burget, L., Matejka, P., Schwarz, P., Glembek, O., and Cernocky, J. (2007). Analysis of feature extraction and channel compensation in GMM speaker recognition system. *IEEE Transactions on Audio, Speech and Language Processing*, 15(7):1979–1986.

[Burget et al., 2009] Burget, L., Matějka, P., Hubeika, V., and Černocký, J. (2009). Investigation into variants of joint factor analysis for speaker recognition. In *Proc. Interspeech 2009*, number 9, pages 1263–1266.

[Burget et al., 2011] Burget, L., Plchot, O., Cumani, S., Glembek, O., Matějka, P., and Brümmer, N. (2011). Discriminatively trained probabilistic linear discriminant analysis for speaker verification. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ.

[Campbell et al., 2006] Campbell, W., Sturim, D., Reynolds, D., and Solomonoff, A. (2006). SVM based speaker verification using a GMM supervector kernel and NAP variability compensation. In *Proceedings of ICASSP 2006*, volume 1, page I.

[Campbell, 2002] Campbell, W. M. (2002). Generalized linear discriminant sequence kernels for speaker recognition. In *Proceedings of Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference*, volume 1, pages I–161 –I–164.

[Cumani et al., 2011] Cumani, S., Brümmer, N., Burget, L., and Laface, P. (2011). Fast discriminative speaker verification in the i-vector space. In *Proc. of ICASSP , 2011*, pages 4852 –4855, Prague, CZ.

[Cumani et al., 2013] Cumani, S., Brümmer, N., Burget, L., Laface, P., Plchot, O., and Vasilakis, V. (2013). Pairwise discriminative speaker verification in the i–vector space. *IEEE Transactions on Audio, Speech and Language Processing*, 21(6):1217–1227.

[Cumani and Laface, 2013] Cumani, S. and Laface, P. (2013). Memory and computation trade-offs for efficient i-vector extraction. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 21(5):934–944.

[Cumani and Laface, 2014a] Cumani, S. and Laface, P. (2014a). Factorized sub-space estimation for fast and memory effective i-vector extraction. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 22(1):248–259.

[Cumani and Laface, 2014b] Cumani, S. and Laface, P. (2014b). Training pairwise support vector machines with large scale datasets. In *Proceedings of ICASSP 2014*, volume 1, pages 1664–1668, Florence, Italy. IEEE-INST ELECTRICAL ELECTRONICS ENGINEERS INC.

[Cumani et al., 2014] Cumani, S., Plchot, O., and Laface, P. (2014). On the use of i-vector posterior distributions in probabilistic linear discriminant analysis. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 22(4):846–857.

[Davis and Mermelstein, 1980] Davis, S. B. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4).

[Dehak et al., 2010a] Dehak, N., Dehak, R., Glass, J., Reynolds, D., and Kenny, P. (2010a). Cosine similarity scoring without score normalization techniques. In *Proc. of Odyssey 2010*, Brno, Czech Republic.

[Dehak et al., 2009] Dehak, N., Dehak, R., Kenny, P., Brümmer, N., Ouellet, P., and Dumouchel, P. (2009). Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification. In *Proc. of the International Conference on Spoken Language Processing (ICSLP)*.

[Dehak et al., 2007] Dehak, N., Dumouchel, P., and Kenny, P. (2007). Modeling prosodic features with Joint Factor Analysis for speaker verification. *Audio*.

[Dehak et al., 2010b] Dehak, N., Kenny, P., Dehak, R., Dumouchel, P., and Ouellet, P. (2010b). Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech and Language Processing*, pages 1 –1.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38.

[Doddington, 1998] Doddington, G. R. (1998). Speaker recognition evaluation methodology: a review and perspective. In *RLA2C Workshop: Speaker Recognition and its Commercial and Forensic Applications*, pages 60–66, Avignon, France.

[Doddington, 2001] Doddington, G. R. (2001). Speaker recognition based on idiolectal differences between speakers. In *INTERSPEECH*, pages 2521–2524, Aalborg, Denmark.

[Ferrer et al., 2011a] Ferrer, L., Bratt, H., Burget, L., Cernocky, H., Glembek, O., Graciarena, M., Lawson, A., Lei, Y., Matejka, P., Plchot, O., and Scheffer, N. (2011a). The PRISM set. `https://code.google.com/p/prism-set/`.

[Ferrer et al., 2011b] Ferrer, L., Bratt, H., Burget, L., Cernocky, H., Glembek, O., Graciarena, M., Lawson, A., Lei, Y., Matejka, P., Plchot, O., and Scheffer, N. (2011b). Promoting robustness for speaker modeling in the community: the PRISM evaluation set. In *Proceedings of SRE11 analysis workshop*, Atlanta.

[Ferrer et al., 2012] Ferrer, L., Burget, L., Plchot, O., and Scheffer, N. (2012). A unified approach for audio characterization and its application to speaker recognition. In *Proceedings of Odyssey 2012*, pages 317–323. International Speech Communication Association.

[Freesound, 2010] Freesound (2010). The freesound database. `http://www.freesound.org`.

[Furui, 1986] Furui, S. (1986). Speaker-independent isolated word recognition using dynamic features of speech spectrum. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 34:52–59.

[Garcia-Romero, 2011] Garcia-Romero, D. (2011). Analysis of i-vector length normalization in Gaussian-PLDA speaker recognition systems. In *Proc. of the International Conference on Spoken Language Processing (ICSLP)*.

[Gauvain and Lee, 1994] Gauvain, J. and Lee, C. H. (1994). Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE Transactions on Speech and Audio Processing*, 2:291–298.

[Glembek, 2012] Glembek, O. (2012). *Optimization of Gaussian Mixture Subspace Models and related scoring algorithms in speaker verification.* PhD thesis, Brno University of Technology.

[Glembek et al., 2011] Glembek, O., Matějka, P., and Burget, L. (2011). Simplification and optimization of i-vector extraction. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ.

[Graff et al., 2001] Graff, D., Miller, D., and Walker, K. (2001). Switchboard cellular part 1 audio. *Linguistic Data Consortium, Philadelphia.*

[Graff et al., 2002] Graff, D., Miller, D., and Walker, K. (2002). Switchboard-2 phase III. *Linguistic Data Consortium, Philadelphia.*

[Graff et al., 2004] Graff, D., Miller, D., and Walker, K. (2004). Switchboard cellular part 2 audio. *Linguistic Data Consortium, Philadelphia.*

[Graff et al., 1999] Graff, D., Walker, K., and Canavan, A. (1999). Switchboard-2 phase II. *Linguistic Data Consortium, Philadelphia.*

[Hatch et al., 2006] Hatch, A. O., Kajarekar, S., and Stolcke, A. (2006). Within-Class Covariance Normalization for SVM-based speaker recognition. In *Proc. ICSLP, Pittsburgh, USA*, pages 1471–1474.

[Hirsch, 2005] Hirsch, G. (2005). Fant - filtering and noise adding tool. `http://dnt.kr.hsnr.de/download.html`.

[Kenny, 2005] Kenny, P. (2005). Joint factor analysis of speaker and session variability: Theory and algorithms - technical report CRIM-06/08-13. Montreal, CRIM, 2005.

[Kenny, 2010] Kenny, P. (2010). Bayesian speaker verification with heavy–tailed priors. In *Proc. of Odyssey 2010*, Brno, Czech Republic. http://www.crim.ca/perso/patrick.kenny, keynote presentation.

[Kenny et al., 2005a] Kenny, P., Boulianne, G., and Dumouchel, P. (2005a). Eigenvoice modeling with sparse training data. *IEEE Trans. Speech and Audio Processing*, 13(3):345–354.

[Kenny et al., 2004] Kenny, P., Boulianne, G., Ouellet, P., and Dumouchel, P. (2004). Speaker adaptation using an eigenphone basis. *IEEE Transactions on Speech and Audio Processing*, 12(6):579–589.

[Kenny et al., 2005b] Kenny, P., Boulianne, G., Ouellet, P., and Dumouchel, P. (2005b). Factor analysis simplified. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 637– 640, Toulouse, France.

[Kenny et al., 2007] Kenny, P., Boulianne, G., Oullet, P., and Dumouchel, P. (2007). Joint factor analysis versus eigenchannels in speaker recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 15(7):2072–2084.

[Kenny and Dumouchel, 2004] Kenny, P. and Dumouchel, P. (2004). Disentangling speaker and channel effects in speaker verification. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, volume 1, pages I – 37–40 vol.1.

[Kenny and Dumouchel, 2004] Kenny, P. and Dumouchel, P. (2004). Experiments in speaker verification using factor analysis likelihood ratios. In *Proceedings of Odyssey 2004*.

[Kenny et al., 2003] Kenny, P., Mihoubi, M., and Dumouchel, P. (2003). New map estimators for speaker recognition. In *INTERSPEECH*.

[Kenny et al., 2008] Kenny, P., Ouellet, P., Dehak, N., Gupta, V., and Dumouchel, P. (2008). A study of inter-speaker variability in speaker verification. *IEEE Transactions on Audio, Speech and Language Processing*, 16(5):980–988.

[Kenny et al., 2013] Kenny, P., Stafylakis, T., Ouellet, P., and Dumouchel, P. (2013). PLDA for speaker verification with utterances of arbitraty duration. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 7649 – 7653.

[Kockmann, 2012] Kockmann, M. (2012). *Subspace Modeling of Prosodic Features for Speaker Verification*. PhD thesis, Brno University of Technology.

[Lin et al., 2008] Lin, C.-J., Weng, R. C., and Keerthi, S. S. (2008). Trust region Newton method for large-scale logistic regression. *Journal of Machine Learning Research*.

[Liu et al., 1989] Liu, D. C., Nocedal, J., Liu, D. C., and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528.

[Lyu et al., 2009] Lyu, S., Simoncelli, E. P., and Hughes, H. (2009). Nonlinear extraction of independent components of natural images using radial gaussianization. *Neural Computation*, pages 1485–1519.

[Martin et al., 1997] Martin, A., Doddington, G., Kamm, T., Ordowski, M., and Przybocki, M. (1997). The DET Curve in Assessment of Detection Task Performance. In *Proc. Eurospeech '97*, pages 1895–1898, Rhodes, Greece.

[Martin et al., 2014] Martin, A. F., Greenberg, C. S., Stanford, V. M., Howard, J. M., Doddington, G. R., and Godfrey, J. J. (2014). Effects of the new testing paradigm of the 2012 nist speaker recognition evaluation. In *Proceedings of Odyssey 2014*, Joensuu, Finland.

[Martínez et al., 2011] Martínez, D. G., Plchot, O., Burget, L., Glembek, O., and Matějka, P. (2011). Language recognition in ivectors space. In *Proc. of Interspeech 2011*, volume 2011, pages 861–864, Florence, IT. International Speech Communication Association.

[Matějka et al., 2006] Matějka, P., Burget, L., Schwarz, P., and Černocký, J. (2006). Brno University of Technology system for NIST 2005 language recognition evaluation. In *Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop*, pages 57–64.

[Matějka et al., 2011] Matějka, P., Glembek, O., Castaldo, F., Alam, J., Plchot, O., Kenny, P., Burget, L., and Černocký, J. (2011). Full-covariance UBM and Heavy-tailed PLDA in i-vector speaker verification. In *Proc. of Interspeech*, Florence, Italy.

[McGovern, 2004] McGovern, S. G. (2004). A model for room acoustics. `http://www.sgm-audio.com/research/rir/rir.html`.

[MITLL, 2103] MITLL (2103). Domain adaptation challenge 2013. `http://www.clsp.jhu.edu/user_uploads/workshops/ws13/DAC_description_v2.pdf`.

[Moore, 2012] Moore, B. C. (2012). *An introduction to the psychology of hearing*. Brill.

[Navrátil et al., 2003] Navrátil, J., Jin, Q., Andrews, W., and Campbell, J. (2003). Phonetic speaker recognition using maximum-likelihood binary-decision tree models. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Hong Kong.

[Nguyen et al., 2000] Nguyen, P., Kuhn, R., Junqua, J.-C., Niedzielski, N., and Wellekens, C. (2000). Eigenvoices: A compact representation of speakers in model space. *Annales des Télécommunications*, 55(3-4):163–171.

[NIST, 2005] NIST (2005). The NIST year 2005 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2005`.

[NIST, 2006] NIST (2006). The NIST year 2006 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2006`.

[NIST, 2008] NIST (2008). The NIST year 2008 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2008`.

[NIST, 2010] NIST (2010). The NIST year 2010 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2010`.

[NIST, 2012] NIST (2012). The NIST year 2012 speaker recognition evaluation plan. `http://nist.gov/itl/iad/mig/upload/NIST_SRE12_evalplan-v17-r1.pdf`.

[Nocedal, 1980] Nocedal, J. (1980). Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation*, 35(151):773–782.

[Nocedal and Nash, 1991] Nocedal, J. and Nash, S. G. (1991). A numerical study of the limited memory BFGS method and the truncated-newton method for large scale optimization. *SIAM Journal on Optimization*, 1(3):358–372.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, 2nd edition.

[Openshaw and Masan, 1994] Openshaw, J. and Masan, J. (1994). On the limitations of cepstral features in noise. In *Proc. ICASSP 1994*, Adelaide, SA, Australia.

[Plchot et al., 2013] Plchot, O., Matsoukas, S., Matějka, P., Dehak, N., Ma, J., Cumani, S., Glembek, O., Heřmanský, H., Mesgarani, N., Soufifar, M. M., Thomas, S., Zhang, B., and Zhou, X. (2013). Developing a speaker identification system for the darpa rats project. In *Proceedings of ICASSP 2013*, pages 6768–6772. IEEE Signal Processing Society.

[Prince and Elder, 2007] Prince, S. J. D. and Elder, J. H. (2007). Probabilistic linear discriminant analysis for inferences about identity. In *11th International Conference on Computer Vision*.

[Rabiner and Juang, 1993] Rabiner, L. and Juang, B.-H. (1993). *Fundamentals of speech recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[Reynolds, 2002] Reynolds, D. (2002). Automatic speaker recognition – acoustics and beyond. JHU SW'02 Tutorial.

[Reynolds, 2003] Reynolds, D. (2003). Channel robust speaker verification via feature mapping. In *Proceedings of ICASSP '03. 2003 IEEE International Conference on*, volume 2, pages II – 53–6 vol.2.

[Reynolds et al., 2000] Reynolds, D., Quatieri, T. F., and Dunn, R. B. (2000). Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, pages 19–41.

[Reynolds and Rose, 1995] Reynolds, D. A. and Rose, R. C. (1995). Robust text-independent speaker identification using gaussian mixture speaker models. *Speech and Audio Processing, IEEE Transactions on*, 3(1):72–83.

[Schwarz et al., 2006] Schwarz, P., Matějka, P., and Černocký, J. (2006). Hierarchical structures of neural networks for phoneme recognition. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 325–328, Toulouse, France.

[Shampine, 2007] Shampine, L. F. (2007). Accurate numerical derivatives in MATLAB. *ACM Trans. Math. Softw.*

[Stevens et al., 1937] Stevens, S. S., Je, and Newman, E. B. (1937). A scale for the measurement of the psychological magnitude of pitch. *Journal of hte Acoustical Society of America*, 8:185–190.

[Torres-Carrasquillo et al., 2002] Torres-Carrasquillo, P. A., Singer, E., Kohler, M. A., and Deller, J. R. (2002). Approaches to language identification using gaussian mixture models and shifted delta cepstral features. In *Proc. ICSLP 2002*, pages 89–92.

[Vapnik, 1995] Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

[Villalba et al., 2013] Villalba, J., Diez, M., Varona, A., and Lleida, E. (2013). Handling recordings acquired simultaneously over multiple channels with plda. In *Proceedings of Interspeech 2013*, Lyon, France.

[Vogt et al., 2005] Vogt, R., Baker, B., and Sridharan, S. (2005). Modelling session variability in text-independent speaker verication. In *Proc. Eurospeech*, pages 3117–3120, Lisbon, Portugal.

[Walker and Strassel, 2012] Walker, K. and Strassel, S. (2012). The RATS radio traffic collection system. In *Proceedings of Odyssey 2012*, Singapore.

[Young et al., 2006] Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X. A., Moore, G., Odell, J., Ollason, D., Povey, D., and et al. (2006). The HTK book.