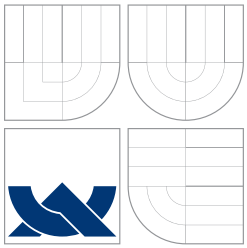
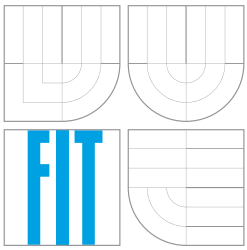


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



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

MODELOVÁNÍ PROZODICKÝCH PŘÍZNAKŮ PRO OVĚŘOVÁNÍ MLUVČÍHO V POD-PROSTORECH

SUBSPACE MODELING OF PROSODIC FEATURES FOR SPEAKER VERIFICATION

DISERTAČNÍ PRÁCE
PHD THESIS

AUTOR PRÁCE
AUTHOR
Dipl.-Ing. MARCEL KOCKMANN

VEDOUCÍ PRÁCE
SUPERVISOR
Doc. Dr. Ing. JAN ČERNOCKÝ

BRNO 2011

Abstract

The thesis investigates into speaker verification by means of prosodic features. This includes an appropriate representation of speech by measurements of pitch, energy and duration of speech sounds. Two diverse parameterization methods are investigated: the first leads to a low-dimensional well-defined set, the second to a large-scale set of heterogeneous prosodic features. The first part of this work concentrates on the development of so called prosodic contour features. Different modeling techniques are developed and investigated, with a special focus on subspace modeling. The second part focuses on a novel subspace modeling technique for the heterogeneous large-scale prosodic features. The model is theoretically derived and experimentally evaluated on official NIST Speaker Recognition Evaluation tasks. Huge improvements over the current state-of-the-art in prosodic speaker verification were obtained. Eventually, a novel fusion method is presented to elegantly combine the two diverse prosodic systems. This technique can also be used to fuse the higher-level systems with a high-performing cepstral system, leading to further significant improvements.

Keywords

speaker verification, prosody, Gaussian mixture models, channel compensation, Joint factor analysis, total variability model, iVector, probabilistic linear discriminant analysis, SNERFs, subspace multinomial model, iVector fusion

Bibliographic citation

Marcel Kockmann: *Subspace modeling of prosodic features for speaker verification*, Doctoral thesis, Brno University of Technology, Faculty of Information Technology, Brno, 2011.

Abstrakt

Předložená disertační práce se zabývá ověřováním mluvčího pomocí prozodických příznaků zahrnujících hodnoty základního tónu, energie a délek řečových úseků. Studovali jsme dvě rozdílné techniky pro parametrizaci: první vede k dobře definované sadě menšího počtu příznaků, druhá k vysoko-dimenzionální sadě heterogenních prozodických příznaků. První část práce se věnuje vývoji příznaků reprezentujících prozodické kontury, zde jsme vyvinuli a ověřili několik modelovacích technik, s důrazem na modelování v reprezentativních pod-prostorech. Druhá část práce se zaměřuje na nové pod-prostorové modelovací techniky pro heterogenní prozodické parametry s velkou dimenzionalitou. Model je teoreticky odvozen a experimentálně ověřen na oficiálních datech z NIST evaluací ověřování mluvčího (NIST Speaker Recognition Evaluation). Ve srovnání s ostatními současnými prozodickými jsme dosáhli podstatně lepších výsledků. Na konci práce presentujeme také novou techniku pro elegantní kombinaci dvou prozodických systémů. Tato technika může být použita rovnvež pro fúzi prozodického systému se standardním přesným cepstrálním systémem, což vede k dalšímu podstatnému zvýšení úspěšnosti verifikace.

Klíčová slova

Ověřování mluvčího, prozodie, modely směsí Gaussovských rozdělení, kompenzace přenosového kanálu, Joint factor analysis, model totální variability, iVector, pravděpodobnostní lineární diskriminační analýza, SNERFs, multinomialní pod-prostorový model, fúze iVectorů.

Bibliografická citace

Marcel Kockmann: *Modelování prozodických příznaků pro ověřování mluvčího v pod-prostorech*, Disertační práce, Vysoké učení technické v Brně, Fakulta informačních technologií, Brno, 2011.

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 Speech@FIT samostatně, nebo ve spolupráci s třetími stranami (Agnitio, CRIM, SRI Internatioinal).

Acknowledgements

First, I would like to thank my former colleagues at Siemens Speech Processing Group, especially Stephan and Bernt. Without their idea to contact Honza for the PhD supervision, I wouldn't have had the opportunity to do my research in such a great speech group.

I would like to thank everybody in Speech@FIT in Brno, especially Lukáš and Honza. Although Honza is the only official supervisor, I have to thank them both equally, Lukas for doing a great job in the technical supervision and Honza for organizing everything and keeping the group running on such a high level. Not to mention all the social events.

Further, I have to thank my colleagues at SVOX, especially Georg who always gave me the freedom to do my research on speaker verification and to attend conferences and workshops.

Also, I thank all the great people I met at conferences and workshops like BOSARIS for the valuable input to my work. Especially, I have to thank Luciana from SRI, for the great collaborative work on prosodic features.

Last but not least, I thank my family and friends for the support and patience they gave me during the last years when I was somewhere between Munich and Brno.

Munich, November 2011

Marcel Kockmann

Contents

List of Abbreviations

ABC	Agnitio-Brno-CRIM
ASR	Automatic Speech Recognition
BFGS	Broyden-Fletcher-Goldfarb-Shanno
BUT	Brno University of Technology
DCF	Detection Cost Function
DCT	Discrete Cosine Transformation
DET	Detection Error Tradeoff
DNA	DesoxyriboNucleic Acid
DP	Dynamic Programming
DTW	Dynamic Time Warping
DVD	Digital Versatile Disc
EER	Equal Error Rate
EM	Expectation-Maximization
F0	Fundamental frequency
FA	Factor Analysis
GD	Gradient Descent
GMM	Gaussian Mixture Model
GSM	Global System for Mobile Communications
HA	Hessian approximation
IDCT	Inverse Discrete Cosine Transformation
IRLS	Iterative Reweighted Least Squares
JFA	Joint Factor Analysis
JHU	Johns Hopkins University
LDA	Linear Discriminant Analysis
LLR	Log-Likelihood Ratio

LR	Logistic Regression
LVCSR	Large-Vocabulary Continuous Speech Recognition
MAP	Maximum-A-Posteriori
MFCC	Mel Frequency Cepstral Coefficients
ML	Maximum Likelihood
NCCF	Normalized Cross-Correlation Function
NERF	New Extraction Region Features
NIST	National Institute of Standards and Technology
NN	Neural Network
PLDA	Probabilistic Linear Discriminant Analysis
RAPT	Robust Algorithm for Pitch Tracking
SGMM	Subspace Gaussian Mixture Model
SMM	Subspace Multinomial Model
SNERFs	Syllable-based Non-uniform Extraction Region Features
SPLDA	Simplified Probabilistic Linear Discriminant Analysis
SRE	Speaker Recognition Evaluations
SRI	Stanford Research Institute
SVM	Support Vector Machine
WCCN	Within-Class Covariance Normalization

1 Introduction

It is well known that it is possible to identify an individual based on its fingerprints. Nowadays, automatic fingerprint verification is a reliable and very accessible technology. Most passports already include digitally stored fingerprints and even commercial products like laptops or DVD-rental machines often use fingerprint scanners for access control.

Besides the unique contour lines of individual fingerprints, there are many more characteristics of a human individual that can be used to specify its identity. Some may be very reliable, like a DNA sequence, but are also very complicated and expensive to extract and analyze. Others, like iris or fingerprint scans may be reliable and cheap, but still need the physical appearance of the individual.

Especially in scenarios where there is only an audio communication channel (like on the telephone), or where hands and eyes are already in use (like when driving a car), speech might be the most preferable source to control devices.

The fact, that there are indeed individual attributes in the human speech signal – similar to those extracted from a fingerprint – is exemplified in Figure 1.1. The picture shows two spectrograms for two utterances with the same content, spoken by two different male adults. At the first sight, high energy regions in different frequency ranges can be observed. This is due to the formant frequencies, based on the shape of the vocal tract of the individual speaker. Their change over time can be observed due to the uttered content and speaking style. Now, by inspecting the lowest high-energy region (the red colored parts), one can already distinguish the two speakers from each other. It can be observed that the first speaker's lowest formant frequency is much lower than the other one.

Attributes like these may be specific for each individual, and by automatically extracting and analyzing many diverse attributes from the speech signal, it is possible to identify an individual solely from its voice.

Automatic extraction and modeling of individual characteristics of a speaker from his or her speech signal is a broad research field these days. The techniques described in this thesis can also be used to extract characteristics from speech other than the speaker identity. It is possible to estimate speaker's age or gender [Kockmann et al., 2010b], the current emotional state [Kockmann et al., 2009, Kockmann et al., 2011a] or the language the individual person speaks [Soufifar et al., 2011].

However, the focus of this thesis lies in the field of speaker recognition, meaning the identification or verification of a speaker solely based on a sample of his or her voice.

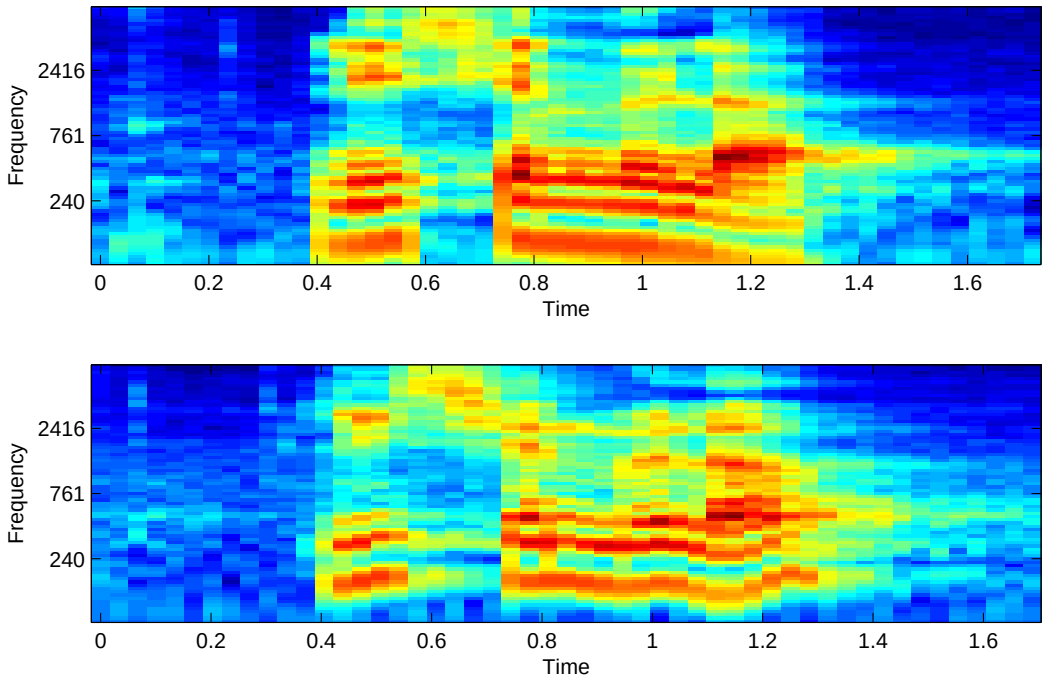
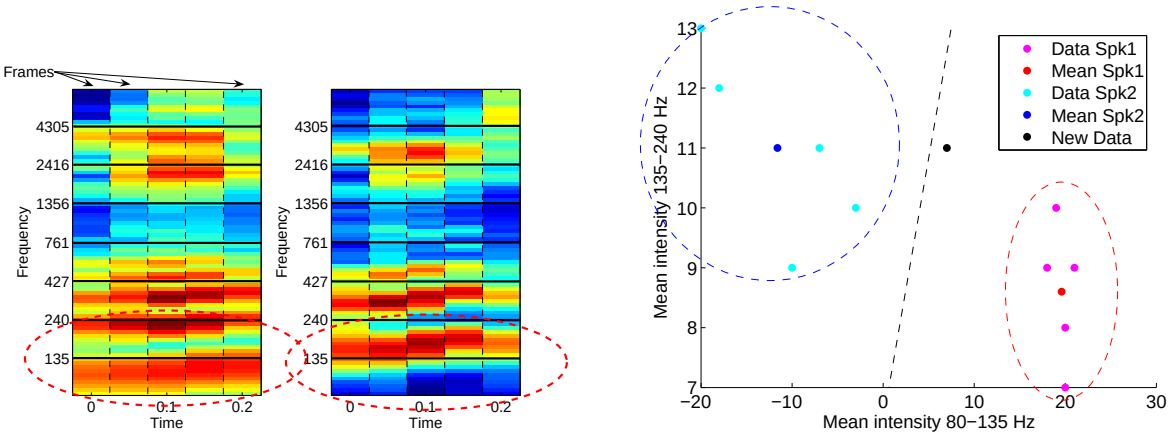


Figure 1.1: Spectrogram for two male speakers uttering the same name.

In speaker recognition one has to further distinguish between speaker identification and speaker verification. Speaker identification is the task of matching an unknown utterance to a set of known speakers. This is usually a classification task with a closed set of classes (speakers in this case). A real world application could be, to identify which known attendee is speaking in a conference call. The amount of possible speakers is known in this case and one has to select the most likely one. In this thesis, the focus lies on the speaker verification task. Here, the task is to verify if a claimed identity matches a speech sample. The task is to either accept or reject the trial consisting of a speech sample and a claimed identity. An application scenario could be access control or telebanking, where the speaker claims his or her identity and the system automatically verifies this claim using the available speech sample.

Further, one has to distinguish between text-dependent and text-independent speaker verification. Text-dependent verification also takes into account what is to be said. An example could be a certain passphrase that is used to unlock a door. In text- independent speaker verification, it is not determined what is said. For example, the text-dependent system might only work if someone uses his name as a passphrase in a certain order. The text-independent should also work if one switches the order of first and last name. However, the main scenario for text-independent speaker verification lies in the intelligence sector, for example to track suspicious persons over intercepted telephone calls.



(a) Feature extraction.

(b) Data in model space.

Figure 1.2: A simple example of automatic speaker recognition.

Exactly this scenario is of interest in the series of the Speaker Recognition Evaluations (SRE), organized by the National Institute of Standards and Technology (NIST) which are ongoing since 1996 [NIST, 1996]. Leading research groups and companies are participating in these evaluations, developing new algorithms and methods to increase the accuracy of automatic speaker verification systems.

1.1 Automatic speaker verification

Speaker recognition, no matter if identification or verification, can be generally split into two phases. Before being able to recognize a speaker by voice, the system has to learn certain characteristics of the voice of an individual person. For this purpose, a supervised *enrollment* phase is needed. One or more speech samples are needed, together with the known identity of the speaker. The automated system will transform the speech signal into appropriate *features* and will usually train a *statistical model* based on these.

A feature extraction unit takes a digitally converted acoustic speech signal as an input and generates feature vectors that represent certain characteristics of the voice in a compressed form. These might be based on the intensity in different frequency ranges of speech, as was already shown using the spectrogram in Figure 1.1. Let us exemplify the feature extraction process by a closer look at the two spectrograms in Figure 1.1. Figure 1.2.a shows a time and frequency quantization of an excerpt of the quasi-continuous spectrum in Figure 1.1. In this simple example, band energies averaged over short time spans – so called frames – are extracted.

These features are extracted for the available training data and are then used to train a compact statistical model representing the speaker, such as a Gaussian Mixture Model (GMM), Logistic Regression (LR) or Support Vector Machine (SVM) (see [Bishop, 2006] for a general introduction on statistical pattern recognition). Usually, only the model

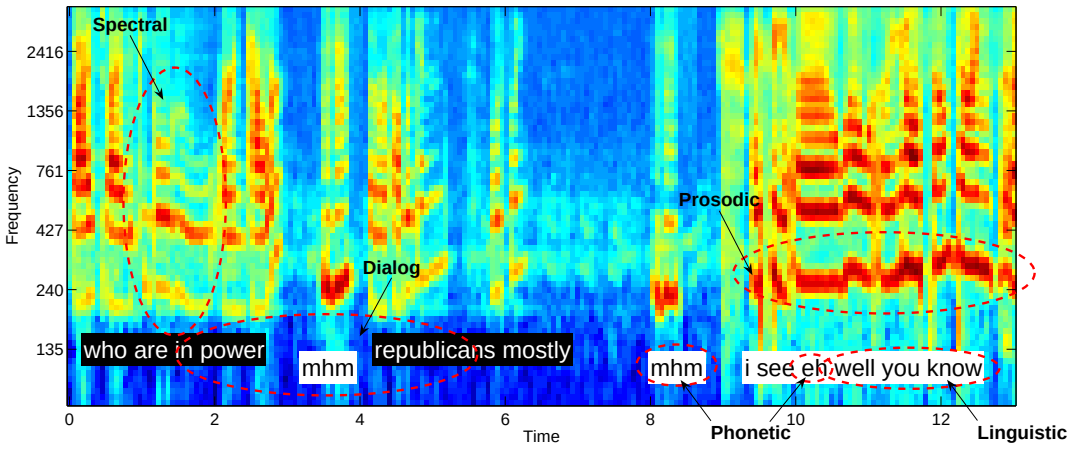


Figure 1.3: Physical and learned attributes of speech.

parameters are kept for the later verification phase. A very simple Gaussian speaker model is depicted in Figure 1.2.b. By taking the five feature frames for the two lowest frequency bands in Figure 1.2.a, a parametric model can be trained for each of the two speakers by computing the mean and the variance of the features in the two-dimensional space. Means are depicted by the blue and red dots and the variances by the dashed circles.

The verification phase always consists of presenting a speech sample and a claimed identity. The speech sample is parameterized by the same feature extraction module as used in the enrollment phase. The system then uses the extracted features to verify whether they come from the claimed speaker or not. For this purpose, it uses the previously trained model parameters for the claimed speaker. The output of the verification system is a probability measure whether or not the speech sample stems from the claimed speaker. Based on this measure, the system will make a decision whether to accept or to reject this hypothesis. Returning to our example in Figure 1.2.b, given the mean and variance parameters, a likelihood can be computed for the new data point (black dot) for both models. In identification, the system would most probably assign the new data to Speaker 2, as the data point produces a higher likelihood given the model parameters of speaker 2. However, for a verification task with a claimed identity for Speaker 2, the system would probably still reject the trial, as the likelihood of the new data point might not exceed the systems acceptance threshold.

1.2 Levels of information in speaker verification

There exist many different cues about speaker's identity which can be extracted from the speech signal. So called low-level cues are determined by physical traits of the voice and higher-level cues depend on traits learned by a speaker. Figure 1.3 shows how the cues at

different levels of speech may be found in the time- or frequency-based progression of the speech signal, or in the lexical content. Ranging from the very low up to the highest levels of speech, several levels of speech cues can be identified [Reynolds, 2002]:

- Spectral level: This is the lowest level and is mainly characterized by the physical traits of the vocal tract.
- Prosodic level: While this level is still based on acoustic traits of the voice, it involves learned habits such as variations in syllable length, loudness and pitch.
- Phonetic level: The cues at this level mainly characterize pronunciation of words adopted by an individual and how different sounds or pauses in speech follow each other.
- Linguistic level: Cues about the identity of a speaker may also be extracted of the used vocabulary in a conversation.
- Dialog level: The cues at this level can only be used within a conversation and is characterized by the behavior during the dialog and how speaker turns appear.

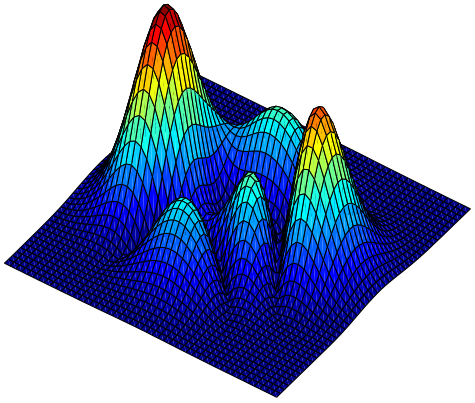
While humans seem to be able to easily use cues from all levels to recognize a certain individual, it becomes more difficult to automatically extract the higher-level cues from the speech signal.

In this thesis, we will focus on the extraction and appropriate modeling of attributes from the prosodic level. While the spectral level seems to be the richest source of information for speaker recognition (yielding the lowest error rates), adding information extracted from speech prosody is an efficient way to improve the overall system performance. This can be done without the need to extract phonetic or linguistic content from even higher levels.

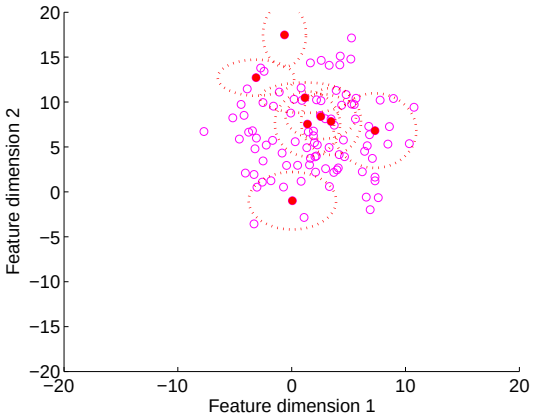
1.3 State-of-the-art

1.3.1 Evolution of low-level speaker verification systems

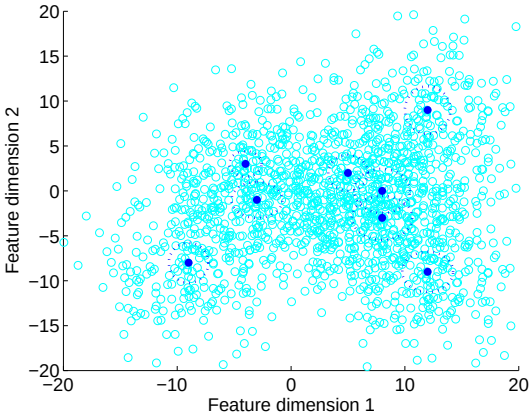
Most automatic speaker verification systems make use of features extracted from the lowest level of speech. They capture purely physical traits of the vocal tract without any higher-level cues, such as intonation, rhythm, stress, speaking style, etc. The speech signal can be seen quasi-static in a time interval below 50ms. Usually, a spectral representation of the speech signal is extracted repeatedly for such short-time windows (see the already introduced examples in Figures 1.1 and 1.2). The most common form of such low-level features in speech processing are Mel Frequency Cepstral Coefficients (MFCC) [Davis and Mermelstein, 1980] which are usually augmented with their first and second order derivatives [Furui, 1986] to capture some temporal context. The typical feature vectors are 40–60 dimensional.



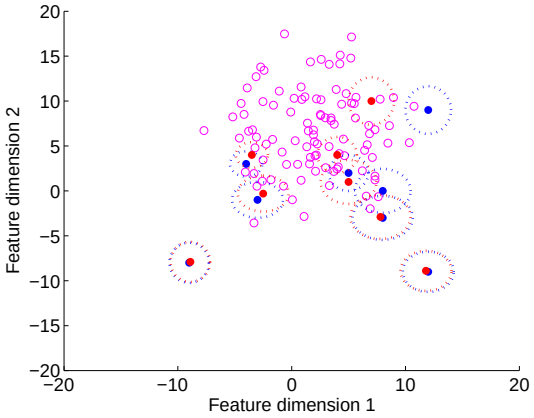
(a) Probability density plot of a GMM.



(b) Model ML estimation.



(c) UBM estimation.



(d) Model MAP estimation.

Figure 1.4: GMM as a basic model for speaker recognition

The foundations for modeling techniques in current state-of-the-art speaker verification systems (see [Kinnunen and Li, 2010] for a detailed review) have been laid more than a decade ago by the introduction of Gaussian Mixture Models (GMM) for text-independent speaker verification [Reynolds and Rose, 1995]. GMMs are parametric models that are able to model complex probability distributions. Figure 1.4.a shows a Gaussian mixture distribution in two dimensions with five components having different weight, mean and variance parameters. These parameters are usually estimated on the training data, using a Maximum-Likelihood paradigm.

A drawback of this approach for small amounts of training data is depicted in Figure 1.4.b. The figure shows some training data in a two-dimensional feature space. The data is used to estimate the means (solid dots) and variances (dashed ellipses) for eight Gaussian components. The main problem is, that there are too many parameters to train from too little data and the model learns the seen data without being able to generalize well for new

unseen data.

Figure 1.4.c shows a more robust model parameter estimation based on much more data from many different speakers. This so called Universal Background Model (UBM) [Reynolds et al., 2000] is used in speaker recognition to cope with the shortcomings of ML estimation. As shown in Figure 1.4.c, the UBM parameters are first estimated on much more data, leading to robust – speaker independent – parameter estimates. The UBM is then used to derive a prior distribution of speaker model parameters. To enroll a speaker model, usually, only the mean parameters are re-estimated using Maximum-a-Posteriori (MAP) adaptation [Gauvain and Lee, 1994]. Figure 1.4.d shows the effect for an adaptation based on the same data as used for the ML update in Figure 1.4.b. The model is only adapted in areas where there is a certain amount of enrollment data available (leading to a ML update for unlimited amount of data), while the mean parameters for unseen data are copied from the UBM.

While this increased the robustness of speaker verification systems, one of the biggest challenges remains: the unwanted variability or channel mismatch between enrollment and verification phase. This can be due to different telephones used, close-talk or hands-free systems, a different room acoustic or background noises. Techniques like score normalization [Auckenthaler et al., 2000] or Feature Warping [Pelecanos and Sridharan, 2001] were successful attempts to compensate for mismatch on the score and feature level. Also Feature Mapping [Reynolds, 2003] works on the feature level, but attempts to map features to a neutral feature space using a model based mapping. A fully model based approach is Speaker Model Synthesis [Teunen et al., 2000], that operates in a similar way to Feature Mapping, but in the model parameter space. Still, both techniques suffer from discrete decisions and data labeling requirements.

[Kenny et al., 2003] used the concept of continuous model based adaptation using subspaces. Figure 1.5 shows the basic idea in a two-dimensional feature space and a single Gaussian component model. The differently colored points represent multiple MAP adapted means from seven different speakers. For each speaker, about twenty different utterances are used to enroll one model per utterance. The solid points represent the means per speaker, from which we can observe directions in which the averaged speaker models differ most. Similarly, we can observe high variability within each colored speaker cluster along the x-axis. By normalizing each speaker cluster with its corresponding mean, we can estimate directions of high session variability. Generalizing this concept to GMMs with many higher-dimensional components, and representing these as supervectors of concatenated mean parameter vectors, it is reasonable to assume that these supervectors mainly live in a much smaller subspace. Using this concept, a supervector with hundreds of thousands of dimensions can be represented efficiently and without loss of discriminative power using a subspace with only a few hundred dimensions. In [Kenny et al., 2003], he used eigenvoice adaptation for rapid speaker adaptation, learning directions of high across-speaker variability. Kenny further proposed to model intersession variability using a low-dimensional latent variable model. Similar attempts to use model based intersession compensation techniques were successfully presented during NIST SRE 2004 [Brümmer, 2004] and the so called *eigenchannel compensation* technique dominated the following SRE 2006

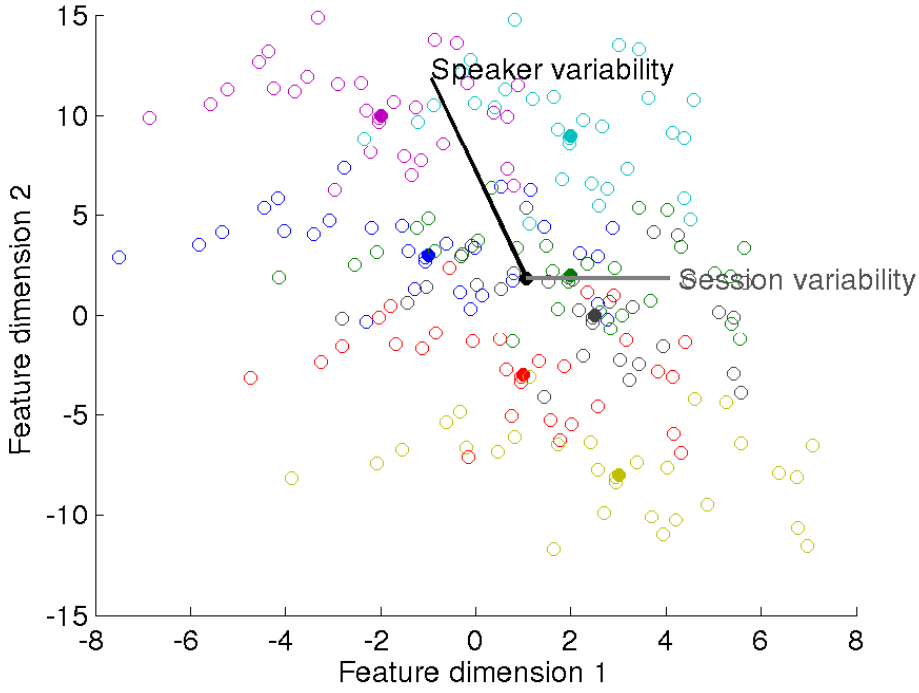


Figure 1.5: Basic idea of subspace-modeling techniques using separate speaker and channel variability.

[Burget et al., 2007]. These techniques learn a low-dimensional subspace within the full space of the GMM mean parameters (so called supervectors) corresponding to directions with high variability due to intersession effects.

The 2008 evaluation was dominated by the Joint Factor Analysis (JFA) paradigm [Kenny et al., 2008b], introducing separate low-dimensional subspaces of speaker and session variability. The JFA model is able to learn the directions of highest speaker- and session variability. This way, rapid adaptation of speaker models even on small amounts of data in combination with a very effective channel compensation became possible.

However, during Johns Hopkins University (JHU) summer workshop on robust speaker verification [Burget et al., 2008] it was found that the assumption of independent channel and speaker subspaces was not optimal and a simplified, but even more effective variant was presented shortly after [Dehak et al., 2009a]. The main difference lies in the total variability modeling with a single low-dimensional subspace representing all the important variability in the space of GMM mean parameters. Furthermore, this way the subspace model is used as a feature extractor by extracting low-dimensional variables, representing utterances. The low-dimensional compact representation of a whole utterance is often referred to as an *iVector*.

Recently, a probabilistic model has been proposed for speaker verification that seems very appropriate to measure the *similarity* between two iVectors, so as to say whether

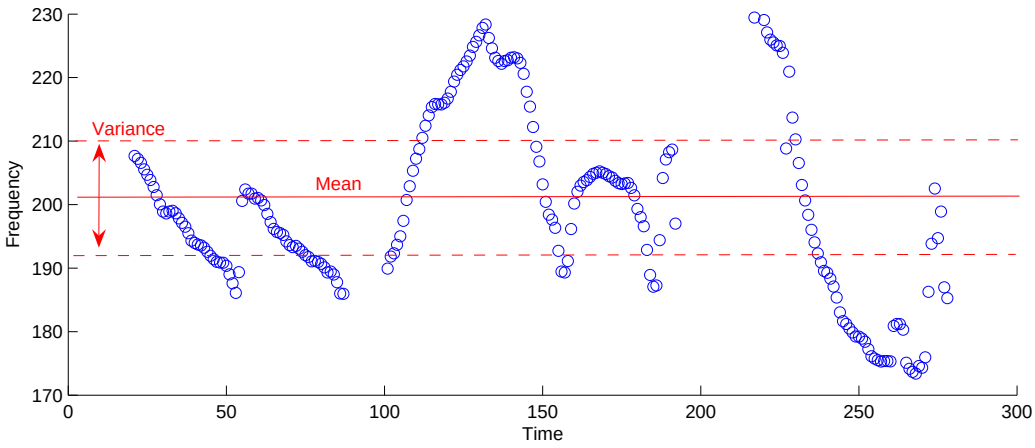


Figure 1.6: Mean and variance of a pitch contour.

two iVectors have been generated by the same speaker or not. This Probabilistic Linear Discriminant Analysis (PLDA) [Prince, 2007] model can efficiently model the speaker and channel variability within the low-dimensional iVector space [Kenny, 2010] and is capable to evaluate speaker trials very efficiently [Burget et al., 2011].

1.3.2 Evolution of prosody-based speaker verification systems

The use of prosody-based features for automatic speaker recognition is known for decades. Some of the very early publications about text-dependent speaker recognition made purely use of fundamental frequency estimates. [Atal, 1972] used an orthogonal transformation of the whole pitch contour of a short utterance as a feature vector. Further, he made use of LDA to consider across- and within-speaker variability and used an Euclidean distance measure for similarity scoring.

In the following years, the use of prosodic cues for speaker characterization was investigated thoroughly [Nolan, 1983, Fant et al., 1990], but with a focus on non-automated applications in forensics. The use of prosodic features in automated systems was rediscovered with launching the NIST speaker recognition evaluations. [Carey et al., 1996] used various statistics of estimated pitch, such as mean and variance, that were estimated over the whole utterance, as exemplified in Figure 1.6. Further, they used an LDA followed by a distance measure, similar to [Atal, 1972]. They first showed that the overall performance of a speaker verification system could be improved by fusion of a cepstral low-level system with a prosodic sub-system on a text-independent task.

During that time, STAR Laboratory at SRI International started their interest in prosodic speaker verification, too. In [Sönmez et al., 1997], a log-normal tied mixture model is proposed to better fit the pitch distribution and to be robust against outliers. Again, fusion with a baseline system was proposed and resulted in significant improve-

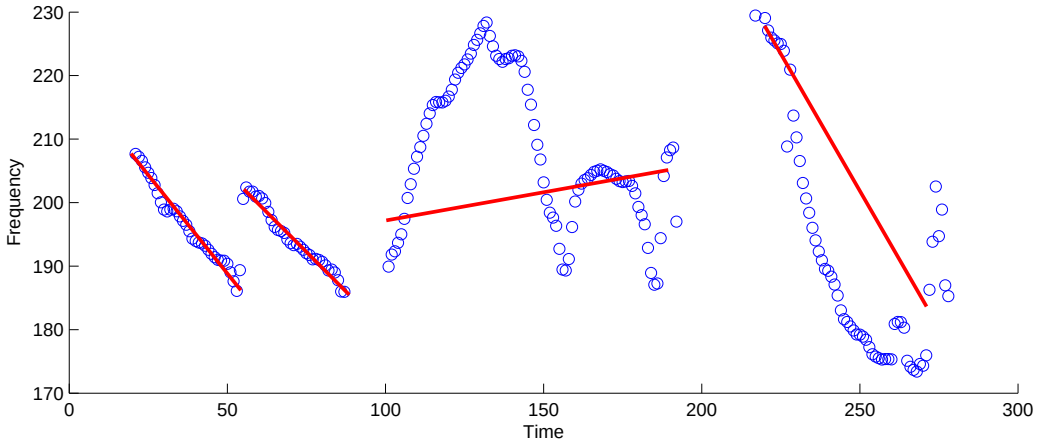


Figure 1.7: Stylized pitch over voiced segments.

ments on the NIST 1996 task. Shortly after, the same authors proposed to incorporate the suprasegmental prosody into the feature extraction process [Sönmez et al., 1998]. They were the first to capture local dynamics in the intonation by fitting a piecewise linear model to the pitch contour. This way, they obtained a stylized pitch contour based on suprasegmental units. This procedure is shown for a pitch contour in Figure 1.7. Further, they included duration information of the voiced units and the lengths of speech pauses into their system. Again, they could obtain improvements of around 10% relative by fusion with a cepstral baseline system on the NIST 1998 task.

Another boost in research on prosodic speaker recognition could be observed after the introduction of the extended data task of the NIST 2001 speaker recognition evaluation, specially conceived to investigate in the use of higher-level features. [Weber et al., 2002] made use of a phone recognizer to build duration models, similar to [Bartkova et al., 2002]. The latter seems to be the first study in which energy measurements were added and a broad investigation into the three main prosodic attributes – duration, pitch and energy – was done. The extended data task (up to 45 minutes of training data per speaker) was also explored in the JHU 2002 Summer Workshop [Peskin et al., 2003]. Many modeling techniques, such as n-gram modeling, k-nearest neighbor, etc., were examined. Especially the work of [Adami et al., 2003] showed promising results. The prosodic baseline system modeled frame-wise pitch and energy measures plus their derivatives in a GMM-UBM framework, as used in cepstral systems. Further, they used n-gram modeling of consecutive, stylized pitch and energy segments (see Figure 1.8) and Dynamic Time Warping (DTW) for pitch contours of selected words.

In 2003, SRI also proposed their new prosodic feature set called New Extraction Region Features (NERF) [Kajarekar et al., 2003]. This was a large set of prosodic features, consisting of several kinds of measurements of duration, energy and pitch, based on various regions of interest, motivated by psycholinguistics. Further work on NERFs

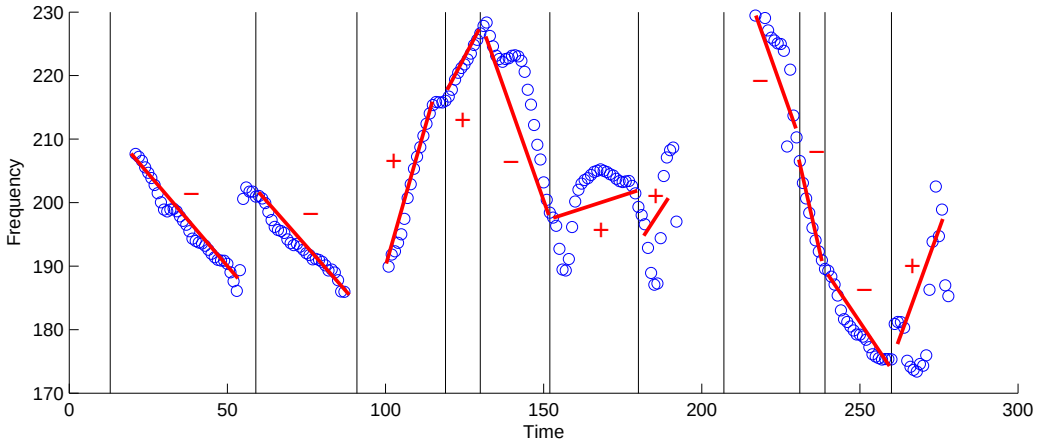


Figure 1.8: Stylized pitch over phonetic segments.

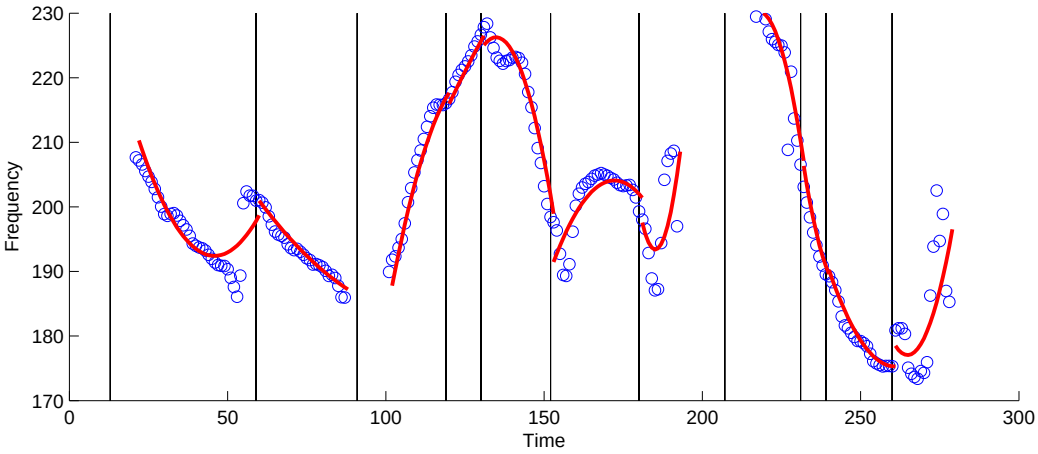


Figure 1.9: Polynomial approximation of pitch contours.

[Kajarekar et al., 2004] addressed the modeling of undefined features and also a new definition of the underlying suprasegmental units [Shriberg et al., 2005]. These syllable-based SNERFs showed very promising results, and in combination with SVM classification [Ferrer et al., 2007] dominated the sector of prosodic speaker verification.

However, due to their high complexity, SNERF modeling was not broadly adopted by the speaker recognition community. Further, intersession variability compensation (as evolved at that time in cepstral modeling, see Section 1.3.1) did not show significant improvements when applied to SNERFs. While most of the early publications on prosodic speaker recognition stated that prosodic features were less affected by noise or channel mismatch, the work by [Dehak et al., 2007] proved that also prosodic features are highly

affected by channel variability. What made this work of special interest was that only a very simple form of prosodic feature set was used, formerly proposed for language identification [Lin and Wang, 2005]. It can be seen as an extension to the linear stylization and segmentation as shown in Figure 1.8. Instead of fitting a linear model to the pitch trajectory, a polynomial curve-fitting algorithm was used to fit a higher order polynomial in a least-square sense to the segment, as shown in Figure 1.9. This way, not only mean and slope of the suprasegmental pitch and energy excerpts are used, but also finer details such as the curvature. Those features were easy to extract and could be used in the JFA framework which was commonly used in cepstral systems. Due to the effective modeling, these simple features performed similar to the much more complex and language-dependent SNERF features.

1.4 Motivation and contribution

After work on phonetic features for speaker verification [Kockmann, 2006] during my Master's thesis, the most promising higher-level approaches seemed to be on the level of speech prosody. The main idea was triggered by some work from the speech synthesis area. [Reichel, 2007] fitted higher order polynomials to pitch segments and built discrete pitch contour classes out of the clustered polynomials. As already mentioned in the last section, I found out very quickly that similar approaches on using continuous approximations of pitch and energy trajectories had already been used in language [Lin and Wang, 2005] and speaker verification. Especially the work on prosodic feature contours for speaker verification [Dehak et al., 2007] motivated me to go on to investigate into this field. The work showed significant improvements in the field of prosodic speaker verification on its own and in combination with a cepstral state-of-the-art system.

1.4.1 Claims of the thesis

The goal of this thesis is to investigate the current state-of-the-art technology for prosodic speaker verification and to further develop feature extraction and modeling techniques to improve the overall accuracy of a combined low- and higher-level speaker verification system. In my opinion, the original contributions are as follows:

- **Prosodic contour features:** I found a compact representation of continuous pitch and energy contours based on suprasegmental units. My proposal was to extract leading Discrete Cosine Transformation coefficients of pitch and energy contours. Suprasegmental units were extracted based on pseudo-syllables created from the output of a phone recognizer. Eventually, I used fixed-size, long-temporal and highly overlapping windows to extract pitch and energy contours.

- **Channel compensation:** I investigated current GMM based approaches for intersession compensation, such as Eigenchannel compensation and Joint Factor Analysis. To my knowledge, I was the first to use total variability modeling of prosodic features.

Investigations were done with a scope on the amount of training data and different channel and speaking style conditions within the NIST evaluations.

- **Channel compensation for SNERFs:** I presented a proposal to transfer the basic idea of subspace-modeling used in JFA to a multinomial model as used to model SNERFs. An iVector front-end was presented to capture the meaningful variability in SNERFs followed by a Probabilistic Linear Discriminant Analysis model.

- **Combination of prosodic systems:** I proposed to use iVector modeling for both, the continuous DCT-based simple contour features and the counts of the discretized SNERFs. A single PLDA model was then trained on the concatenated iVectors; this is called iVector fusion.

- **Comparative study:** I presented a thorough comparison of the current state-of-the-art systems and the proposed systems on the latest NIST speaker recognition evaluation datasets, including different channel and speaking style conditions.

- **Fusion with cepstral system:** Finally, I have shown gains achieved when fusing the proposed methods with best performing cepstral system. Again, an iVector fusion approach is proposed.

1.4.2 Content of the thesis

The document is organized as follows:

Chapter 2 introduces the databases and evaluation metrics which will be used through this thesis. This is done first, in order to familiarize the reader with the experimental conditions that are presented in all following chapters. Note, that the test environments may change during the work, due to the ongoing process of NIST evaluations.

Chapter 3 presents the investigated and further developed parameterization methods of speech for prosodic speaker verification. Beside the theoretical approaches for pitch and energy approximation and segmentation, I will also present experimental results that have been published during the work on this thesis.

Chapter 4 has a more theoretical character, as it was of great desire to understand the modeling approaches for speaker and session variability modeling to be able to derive a new approach applicable to highly complex prosodic features. Experimental results are also presented that confirm the effectivity of the proposed algorithms.

Chapter 5 presents a final comparative study to compare and combine the most promising approaches derived in this thesis and to further fulfill the main claim of improving the overall performance of the speaker verification system due to the use of additional prosodic information.

Chapter 6 contains the conclusions drawn from the work and points out directions for future research.

Generally, only equations needed to implement the used algorithms are given through the thesis. Detailed derivations of the used modeling techniques can be found in the Appendix.

2 Evaluation metrics and data

The metrics and databases used to evaluate the techniques described in this thesis are presented first. This is done at this early stage in order to familiarize the reader with the terms used from Chapter 3 on. All experimental results are presented on official NIST Speaker Recognition Evaluation tasks[1] using metrics defined by NIST. We will refer to results obtained and published over a time span of four years, including two NIST evaluations. Due to this fact, early experiments are performed on the official core conditions of NIST SRE 2006 [NIST, 2006], followed by experiments on more recent NIST 2008 evaluation [NIST, 2008] and eventually also on the current NIST 2010 evaluation [NIST, 2010]. Parallel, also a shift in the primary evaluation metric happened, emphasizing more and more the need for systems producing less false acceptances.

2.1 Evaluation metrics

When it comes to evaluation of the system performance, one has to distinguish clearly between the task of speaker identification and speaker verification. A speaker verification system always has to decide if the claimed identity matches the speech utterance under test. In speaker verification, there are always two kinds of errors, so there is no single quality measure such as one error rate. A *false rejection* describes the error made by rejecting a true speaker, while a *false acceptance* stands for an error made by the system by accepting an impostor speaker. The amounts of these errors are correlated and depend on the operating point of the speaker verification system. A very high threshold might never accept an impostor speaker, but most likely also the true speaker will rarely be accepted. Setting a threshold will always depend on the requirements of a certain use-case and will be a tradeoff between security and usability.

2.1.1 DET plots

A generic tool to validate the quality of automatic speaker verification systems are the Detection Error Tradeoff (DET) plots. Figure 2.1 shows an example of DET plots for two different systems. The x-axis gives the probabilities of false acceptances, while the y-axis

[1] `http://www.itl.nist.gov/iad/mig//tests/sre`

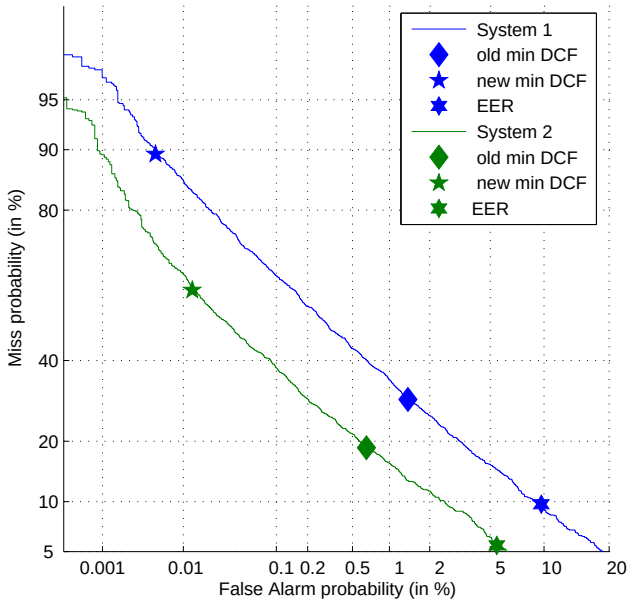


Figure 2.1: DET curves for two different systems. The three markers in each line correspond to the new DCF, the old DCF, and the EER, from left to right.

stands for the probabilities of false rejections, derived from the scores of an evaluation database. The DET plot gives an overall impression of the system quality for all operating points, from very low false acceptances to very low false rejections. In this example, the quality of System 2 is generally better than System 1, as the whole line moves closer to the left-bottom corner of the plot.

2.1.2 Equal Error Rate

To simplify the comparison of different systems, it is common to select specific points from the DET plot. A very popular and intuitive measure is the Equal Error Rate (EER) which is the point on the DET curve where the probability of false acceptances and false rejections is equal. In Figure 2.1, the EER is shown by the right markers.

2.1.3 Detection Cost Function

As the EER usually does not correspond to the needs of real speaker verification systems, NIST has introduced a different metric as the primary measure for its *Speaker Recognition Evaluation* series. Usually, false acceptances are much more "expensive" than false rejections in a real application. Just imagine a thief getting access to a bank account compared to the situation where the true client has to verify himself one more time. For this

Type	C_{Miss}	$C_{\text{FalseAcceptance}}$	P_{Target}
old DCF	10	1	0.01
new DCF	1	1	0.001

Table 2.1: Speaker detection cost model parameters used until SRE 2008 (*old DCF*) and for SRE 2010 (*new DCF*).

purpose, NIST uses a cost metric called Detection Cost Function (DCF), which measure speaker verification performance at a specific operating point. DCF directly considers the overall costs based on the false acceptances and false rejections made by the system. This detection cost function is defined as a weighted sum of miss and false acceptance error probabilities:

$$C_{\text{Det}} = C_{\text{Miss}} \times P_{\text{Miss|Target}} \times P_{\text{Target}} + C_{\text{FalseAcceptance}} \times P_{\text{FalseAcceptance|NonTarget}} \times (1 - P_{\text{Target}}). \quad (2.1)$$

The parameters of this cost function are the relative costs of detection errors, C_{Miss} and $C_{\text{FalseAcceptance}}$, and the a-priori probability of the specified target speaker, P_{Target}. The common values of these parameters are given in Table 2.1. For the SRE 2010, NIST introduced a new parameter set to emphasize the importance of very low false acceptances. Furthermore, DCF is usually normalized by dividing it by the best cost that could be obtained without processing the input data (i.e., by either always accepting or always rejecting the segment speaker as matching the target speaker, whichever gives the lower cost):

$$C_{\text{Default}} = \min \begin{cases} C_{\text{Miss}} \times P_{\text{Target}} \\ C_{\text{FalseAcceptance}} \times (1 - P_{\text{Target}}) \end{cases} \quad (2.2)$$

$$C_{\text{Norm}} = C_{\text{Det}} / C_{\text{Default}} \quad (2.3)$$

As the cost depends on the actual decisions that are made based on the set threshold, one can further distinguish between two types of DCF. The *min DCF* is the minimum that can be achieved if the threshold is set optimal for the given test data. However, for a real application, the decision threshold has to be set in advance and new data is processed using this parameter. The *act DCF* therefore reflects the actual cost that a system achieves on new data with a threshold decided on development data.

DCF points can also be visualized on the DET curve. Figure 2.1 shows the new and the old minimum DCF points. It can be observed that the new DCF moves the desired operating point much further in the area of very low false acceptances.

2.2 Databases

Generally, there are three kinds of data that are involved in the process of automatic speaker recognition:

- **Background set:** A huge dataset containing speech from many speakers (in the order of thousands) from the expected target population. Initially used to train the UBM, it is also used to train the subspace models or to draw impostor models for score normalization. The used databases and their application are described in Sections 2.2.1 and 2.2.2.
- **Development set:** An evaluation database that can be used to optimize parameter settings, containing training and test utterances for many speakers. Sections 2.2.2 and 2.2.3 will describe the datasets used through this work.
- **Evaluation set:** Another independent evaluation database that is used to evaluate the final system (that was optimized on the development set). The dataset as described in Section 2.2.4 will be used for the final experiments.

2.2.1 Switchboard, NIST SRE 2004 and 2005

The databases described here are not involved in any verification process. They are mainly used to train background models or to draw impostor sessions for normalization. As they contain many recordings from the same speakers over diverse channels, they are especially attractive to estimate characteristics of intersession variability. The Switchboard series comprises several releases recorded from the early 90's until 2004. Four of these are included in the training setup:

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] was 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 landline 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.

The NIST SRE 2004 corpus [Martin and Przybocki, 2004] consists of 10,743 telephone call segments recorded from 480 participants (181 Male, 299 Female) over landline as well as cellular phones. The NIST SRE 2005 [NIST, 2005] corpus consists of 16,537 telephone call segments recorded from 528 participants (220 Male, 308 Female) over landline as well as cellular phones. Additionally, telephone calls were recorded over auxiliary microphones of eight different kinds. For both NIST corpora, many segments have different lengths (from

10 seconds up to five minutes) but may stem from the same original full conversation. Furthermore, some segments contain summed conversations. Only unique full conversations with separate channel per speaker are used in the setup. Apart from native speakers, both collections also consist of non-native English and several foreign languages.

2.2.2 NIST SRE 2006

This corpus is used for many early experiments that are reported during the thesis. However, for experiments on NIST SRE 2008 and 2010 the corpus has also been included into the background data set.

Overall, the NIST SRE 2006 corpus [NIST, 2006] consists of 24,637 telephone call segments recorded from 1088 participants (462 Male, 626 Female) over landline as well as cellular phones. Additionally, telephone calls were recorded over auxiliary microphones of eight different kinds. Again, many segments have different lengths (from 10 seconds up to five minutes) but may stem from the same original full conversation. Furthermore, some segments contain summed conversations. Only unique full conversations with separate channel per speaker are used in the setup. Again, native as well as non-native English and several foreign languages are recorded. Special attention has to be paid while using this data, as recordings from the NIST SRE 2005 corpus have been recycled.

Experiments that report results on the NIST 2006 corpus are always performed on the core condition which contains English trials only. The 1-side training 1-side test condition is considered, where approximately 2.5 minutes of speech (from a 5-minute telephone conversation) are available to train each speaker and for each test utterance. This set contains 329 female and 248 male training utterances (multiple utterances can be produced by one distinct speaker), 1,846 target trials, and 21,841 nontarget trials.

2.2.3 NIST SRE 2008

In the 2008 evaluation [NIST, 2008], NIST broadened the scope of the evaluation by introducing interview speech that was recorded over several microphones. As a consequence, even the core condition (only full five minute calls in English speech) contains different subconditions involving different types of speech or channels during both speaker enrollment and verification. During the thesis, the results on the 2008 corpus are reported for the following conditions: *tel-phn:tel-phn* uses only conversational telephone speech of full calls for enrollment and verification, with 1,154 target and 1,516,837 nontarget trials (equivalent to the preceding years). *int-mic:tel-phn* uses interview speech recorded over several microphone types for enrollment and conversational telephone calls for verification, with 1,459 target and 820,215 nontarget trials. The condition *int-mic:int-mic* uses interview speech recorded over microphone for both enrollment and verification, consisting of 33,743 target and 1,108,882 nontarget trials.

Note, that the original NIST tasks are extended to include about two orders of magnitude more impostor samples. This was done to support the new DCF metric introduced by NIST for the 2010 evaluation [NIST, 2010]. Furthermore, a held-out set of 67 speakers

was defined to be included into the background set. This was done in order to overcome the shortage of interview data in the background set. A detailed description of the task definition can be found in [Scheffer et al., 2010].

2.2.4 NIST SRE 2010

Finally, results are reported on selected conditions of the NIST 2010 extended evaluation [NIST, 2010], that match the conditions in the 2008 development set: *tel-phn:tel-phn* uses only conversational telephone speech of full calls for enrollment and verification with 7,169 target and 408,950 nontarget trials (official extended condition 5). *int-mic:tel-phn* uses interview speech recorded over several microphone types for enrollment and conversational telephone calls for verification with 3,989 target and 637,850 nontarget trials (official extended condition 3). The condition *int-mic:int-mic* uses interview speech recorded over microphones for both enrollment and verification, consisting of 15,084 target and 2,789,534 nontarget trials (official extended condition 2).

3

Parameterization of speech for prosodic speaker verification

Acoustic attributes of speech prosody mainly involve variations in syllable length, loudness and pitch. Prosodic features are not confined to small segments of speech, but cover longer time-spans. For this reason, speech prosody is often said to be suprasegmental. Those prosodic units need not to correspond to grammatical units, though they can, and often syllables are chosen as the underlying units.

To extract prosodic features from speech, three main modules are needed:

1. Loudness measure: As a measure of loudness, the short-term energy of the speech signal is usually extracted directly from the signal or from the spectrum.

2. Pitch measure: Several algorithms exist to estimate the fundamental frequency from the speech signal, many are based on the cross-correlation of the time signal indicating periodic (voiced) regions.

3. Prosodic units: The underlying units to model the suprasegmental character of speech prosody are usually determined by even higher-levels of speech representation derived from a phone recognizer or even a full Automatic Speech Recognition (ASR) system.

3.1 Prosodic contour features

As mentioned in the introduction, the initial intention was to use a finer modeling of pitch and energy contours than used in the linear stylization by [Sönmez et al., 1998] and [Adami et al., 2003] as depicted in Figure 1.7. The use of a curve-fitting algorithm based on higher-order polynomials [Reichel, 2007] seemed to be an appropriate way, suitable also for speaker recognition. This way, each pitch or energy segment can be represented by a fixed number of the corresponding polynomial coefficients and form a fixed-sized feature vector. It is then possible to model these prosodic feature vectors by standard UBM-GMM paradigm [Reynolds et al., 2000] as used for standard cepstral-based features.

In the very early literature research phase of this thesis, it was found that the same idea was recently implemented by [Dehak et al., 2007]. Not only did they use a polynomial

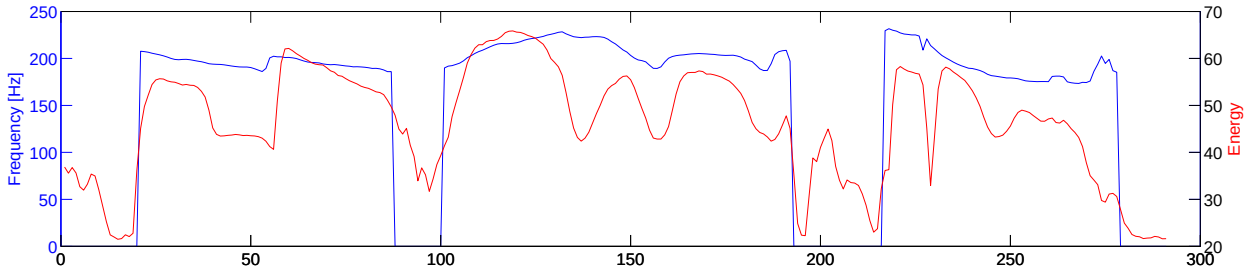


Figure 3.1: Pitch (blue) and energy (red) contours extracted for a whole sentence.

approximation of pitch and energy based on suprasegmental units, but also they already incorporated intersession variability compensation based on Joint Factor Analysis in the modeling approach.

Although this idea of curve-fitting based prosodic feature extraction has already been used and published, the excellent results obtained by [Dehak et al., 2007] motivated me to continue the work on the prosodic level and to develop an own prosodic feature extraction module. The proposed approach to prosodic contour feature extraction differs in two points:

1. First, another way of parameterizing the temporal trajectories is used. [Dehak et al., 2007] used Legendre polynomials that are fitted in a least-square-error sense to the original contour segments. Here it is proposed to use Discrete Cosine Transformation of each variable lengths pitch and energy segment.

2. The second idea was to derive the suprasegmental units in a different way. In the work by [Dehak et al., 2007], the segmentation is simply based on local minima in the signal energy. On the one hand, in my proposed approach, even higher level information should be incorporated, by deriving pseudo-syllable units using a language-independent phone recognizer. On the other hand, we also tested a very simple fixed-size long-temporal context.

This section describes the process of extracting the proposed prosodic contour features. First, we outline how loudness and fundamental frequency measures are obtained. Next, we present how the duration measure is obtained by segmenting the speech in suprasegmental units. Finally, we show how to parameterize the information encoded in loudness and fundamental frequency for each variable-sized suprasegmental unit to a fixed-sized feature vector.

3.1.1 Basic prosodic features

Pitch

The quantity that is actually being estimated by all "pitch trackers" is the fundamental frequency (F0). F0 is defined as the lowest frequency of a periodic waveform and is an

inherent property of periodic speech signals. It tends to correlate well with perceived pitch (that is strictly defined otherwise, see [Talkin, 1995]). In time domain, it can be defined as the inverse of the smallest period in the interval being analyzed. For typical male adults, F0 will lie between 85–180 Hz and for females between 165–255Hz [Titze, 2000].

We will briefly describe a popular family of pitch algorithms that work directly on the time signal [Talkin, 1995]. Those F0 estimation algorithms comprise three stages:

1. Pre-processing.

2. Estimation of candidates for true periods.

3. Selection of the best candidate and F0 refinement.

The aim of the pre-processing phase is to remove interfering signal components from the audio signal. This is usually done by a band-pass filter or some sort of noise reduction. Note, that a standard telephone signal (that we mostly work with) is already band-pass filtered from 300–3400Hz due to the standard telephone channel. However, the fundamental frequency can still be inferred through its harmonics in the signal.

The estimation of F0 candidates itself is mostly performed directly on the time signal using correlations within the signal as a traditional source of period candidates. A widely used and robust pitch tracking algorithm is the Robust Algorithm for Pitch Tracking (RAPT) algorithm [Talkin, 1995], that is based on the Normalized Cross-Correlation Function (NCCF). It consists of the following steps:

1. Generate two version of sampled speech data, one at the original sample rate and one at a significantly reduced rate.

2. Compute NCCF of low sample rate signal for all lags in the F0 range of interest. This first pass records the located local maxima.

3. Compute NCCF of high sample rate only in vicinity of the peaks found in the first pass, again record new maxima.

4. Generate F0 candidates and unvoiced probability for each frame from the second NCCF pass.

5. Use Dynamic Programming (DP) to select the best path through the candidates of the whole utterance.

The output of a pitch tracker is a continuous F0 contour. The blue line in Figure 3.1 shows pitch values estimated every 10ms for a whole sentence with the RAPT algorithm. When there is no pitch detected (in unvoiced regions or speech pauses) the algorithms simply outputs zeros.

Energy

Prosodic features measuring the loudness of speech are usually directly obtained from the signal energy [Bartkova et al., 2002]. The short-time energy of the speech signal can be either extracted directly from the time signal or equivalently from its squared magnitude spectrum. As shown by the red line in Figure 3.1, the log-energy measure is also extracted in 10ms steps.

Post-processing

Before any further processing, the raw pitch and energy values are first transformed to the logarithmic domain to compress their dynamic range. The energy values are further normalized by subtracting the maximum value over the whole utterance to make the loudness measure less dependent on channel effects or the amplification of the signal. The pitch values are further filtered by a median filter to smooth the contour.

3.1.2 Suprasegmental units

The time span of the prosodic suprasegmental units is used in two ways for the contour features: First, the size of each segment is used as a single duration feature. Second, the segment boundaries determine the pitch or energy sequence that is being modeled.

The literature proposes many methods to define suprasegmental units for prosodic feature extraction, most of them using phonetically motivated syllable-like units. A syllable can be seen as a unit of organization of speech sounds, or as a phonological building block which has influence on rhythm, stress and other prosodic attributes of speech.

Various approaches will be investigated, with a special interest in their computational complexity and further constraints, such as language dependence. Two of these approaches have been proposed during the work on this thesis and are described in the following.

Pseudo-syllable segmentation

The first approach to segment the speech into syllable-like units is based on the basic assumption that a syllable is typically made up of a syllable nucleus (most often a vowel) with optional initial and final margins (onset and coda, typically consonants). By using this assumption we can derive syllable-like units from a phone recognizer. Further, to be less language dependent, one can use a phone recognizer with a high number of phones, for example the Hungarian recognizer from BUT [Schwarz et al., 2006]. The proposed segmentation algorithm consists of the following steps:

1. Extract Hungarian phones.
2. Map phones to coarse classes 'silence', 'vowel' and 'consonant'.
3. For each region between two silence labels:

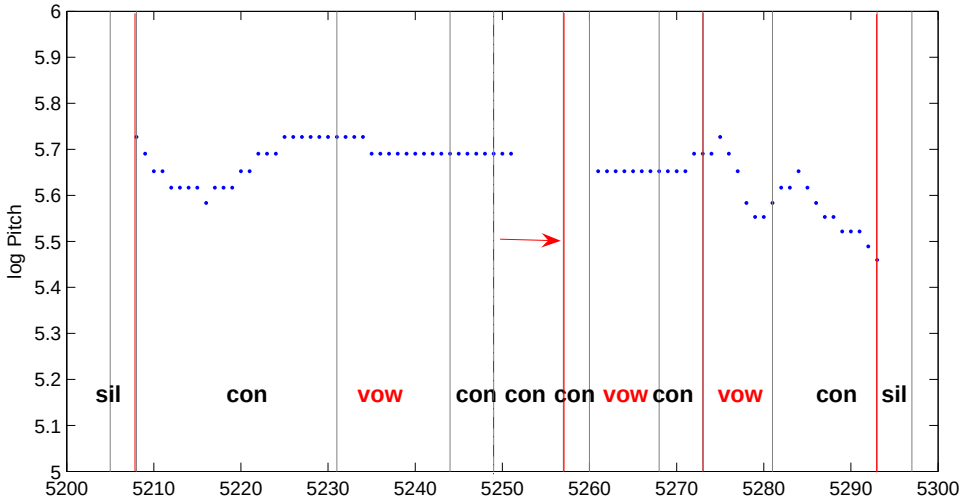


Figure 3.2: Pseudo-syllable generation from vowels and consonants. Each vowel is considered as the nucleus of a syllable, preceding consonants as onset and successive consonants as coda.

- Consider each vowel as the nucleus of a syllable.
- Set the syllable boundaries to the phone boundaries that are the closest to the points right in the middle between two vowels.
- If a syllable boundary lies in the middle of a sequence of voiced frames, while another possible candidate does not, move the boundary there.

This process is illustrated in Figure 3.2. The vertical lines indicate the phone boundaries. Highlighted are the three vowels that are found for a speech segment between two pauses. Next, the algorithm tries to set the syllable boundaries equidistantly between the vowels. As there are three consonants between the first and the second vowel, the algorithm picks the first consonant boundary (near frame 5250) instead of the second. However, the successive processing stage finds that there is a continuous pitch contour that would be cut by this segmentation, while there is no pitch detected at the boundary of the second consonant. The syllable boundary is therefore shifted (indicated by the red arrow). The length of the obtained syllable segment is also used as a single duration feature.

Fixed-size segmentation

While the previous algorithm itself is quite simple, it still needs a complex phone recognizer incorporating cepstral features. As a second approach, it is proposed to simply model the contours of pitch and energy over a fixed-size window. As this segmentation does not rely on any data-driven assumptions where to define the suprasegmental units, it works with highly overlapping windows and a window size that corresponds to an estimated average syllable length. This way, highly correlated and maybe redundant feature frames are generated, many more than for the non-overlapping and exclusive segmentation in

the former approach. As this approach is somehow similar to the extraction of MFCC with a fixed and overlapping analysis window, it is expected that the successive statistical modeling technique of GMMs will learn the relevant information and will be able to benefit from the increased number of features per utterance.

In this case, the number of voiced frames within the analysis window is used as a duration feature.

3.1.3 Contour approximation

Eventually, the extracted pitch and energy measures should be represented in the context of each suprasegmental unit. To be able to feed these prosodic features to a statistical model such as a GMM, a fixed-size representation for each variable-sized suprasegmental unit is needed. For this purpose, a curve-fitting algorithm seems appropriate that best fits a combination of different polynomials of different degrees to the original trajectory in a least-squared-error sense. This way, it can capture the continuous contour by simply keeping the coefficients corresponding to the polynomial basis functions.

In [Lin and Wang, 2005], it is proposed to fit the energy and pitch contours extracted over a suprasegmental unit by a curve fitting algorithm based on Legendre polynomials [Abramowitz and Stegun, 1972]. The advantage over simpler polynomials is, that they are defined by orthogonal basis functions. As the Legendre polynomial is only defined in the interval of -1 to 1, all pitch and energy measures for the suprasegmental units need to be mapped to this interval first.

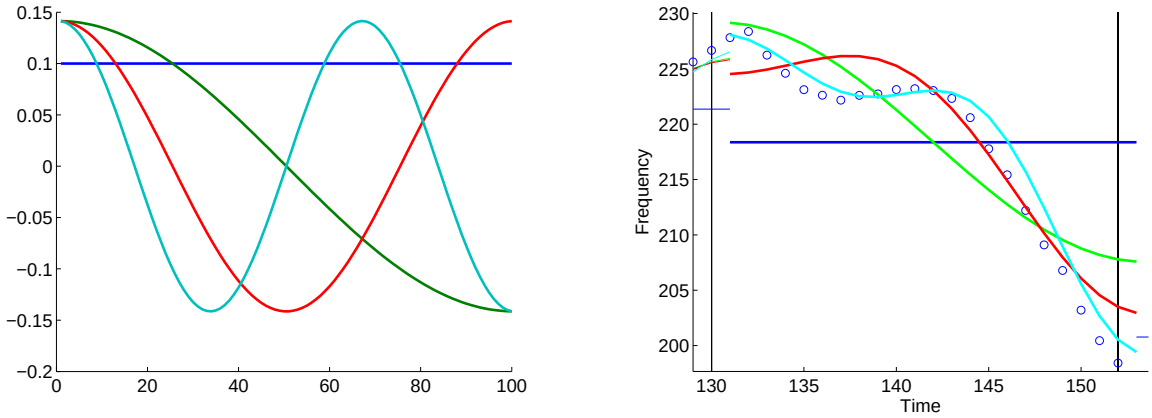
Here, we propose to simply apply Discrete Cosine Transformation (DCT) to the extracted pitch and energy values $x(n)$ extracted for each suprasegmental unit of length N:

$$y(k) = w(k) \sum_{n=1}^{N} x(n) \cos\left(\frac{\pi(2n-1)(k-1)}{2N}\right) \quad k = 1, 2, \ldots N, \tag{3.1}$$

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}} & k = 1 \\ \sqrt{\frac{2}{N}} & 2 \leq k \leq N \end{cases} . \tag{3.2}$$

Taking the Inverse Discrete Cosine Transformation (IDCT) of all coefficients $y(k)$ would result in perfect reconstruction of each pitch or energy contour extracted for each variable sized suprasegmental unit. However, taking only a fixed number of the leading DCT coefficients results in an approximated curve for each segment. This is illustrated in Figure 3.3: Figure 3.3.a shows the first four orthogonal DCT basis functions that are used to transform the original pitch and energy values. Figure 3.3.b shows an excerpt of the pitch contour already shown in the introduction in Figure 1.6. The solid lines show how the contours can be approximated by using only the first (blue) up to the first four (cyan) DCT coefficients.

This way, each variable-sized pitch or energy contour can be translated to a fixed-sized parametric representation. Similar to the Legendre polynomials, the coefficients correspond to the mean, slope, curvature and fine details of the original contour. This becomes clear



(a) DCT basis functions

(b) Contour approximation by DCT

Figure 3.3: Approximation of pitch contour by the first four DCT basis functions.

when observing the first DCT basis as plotted in Figure 3.3.a. Note, that unvoiced frames (where no pitch is detected) need to be treated specially. Some methods will be described in the experimental section.

3.1.4 Final feature vector

The final feature frames are constructed per syllable-like unit in the utterance. The segmentation boundaries determine the length of the segment which is stored in the feature vector as a single discrete number. Next, the first n DCT coefficients are stored for the pitch as well as for the energy contour. So, for each syllable, we obtain a $2n+1$ dimensional feature vector.

3.1.5 Experiments

This section presents selected experiments, using different configurations of the proposed contour features, that have also been published during the work on this thesis.

Duration, Pitch and Energy

In [Kockmann and Burget, 2008a], the first experiments using the DCT contour features as described in the last section were presented. The modeling was based on standard UBM-GMM paradigm for speaker verification with MAP adaptation [Reynolds et al., 2000]. Results are reported for the 2006 NIST SRE condition as described in Chapter 2.2.2.

For the prosodic feature vectors, the DCT approximation as described in Section 3.1.3 was used and the pseudo-syllable suprasegmental units were derived using the algorithm described in Section 3.1.2. Only voiced frames within each segment are used (unvoiced frames are simply cut out), determined by the pitch tracker. The phonetic alignments

Feature Vector	Dim	EER [%]
Pitch Contour	6	29.67
Duration, Pitch Contour	7	29.1
Pitch & Energy Contour	12	28.37
Duration, Pitch & Energy Contour	13	**25.73**

Table 3.1: Different prosodic feature vectors with 6 DCT coefficients per contour. NIST SRE 2006 core condition. Relevance MAP GMM-UBM system.

were obtained using a neural network based Hungarian phone recognizer with long temporal context [Schwarz et al., 2006, Schwarz, 2009, Schwarz et al., 2008].

The final prosodic feature vectors are extracted per suprasegmental unit without overlap and consist of the DCT coefficients for the pitch and the energy contour. Furthermore, the duration of the extracted pseudo-syllable is appended to the feature vector.

These features are first extracted for the whole background training data (see Chapter 2.2.1) and two gender-dependent UBMs with 256 Gaussian components and diagonal covariance matrices are trained using standard EM algorithm [Dempster et al., 1977]. Speaker models are derived by standard relevance MAP adaptation [Reynolds et al., 2000] of the mean parameters with a relevance factor $\tau = 16$. Scoring is based on a Log Likelihood Ratio (LLR) between the speaker model and the UBM.

The first experiments aimed at investigating the importance of the individual prosodic components — duration, pitch and energy — and the general performance of the proposed contour features.

Table 3.1 shows results for different sets of prosodic feature vectors, from using only the pitch contour up to duration, pitch and energy features. The best results are achieved by using 13-dimensional vectors comprising one value for duration and 6 DCT coefficients for pitch and energy contour each.

Segmentation and contour modeling

Next, the effect of the degree of smoothing of the real pitch and energy contours due to the use of only n leading DCT coefficients was investigated. Table 3.2 shows the results for varying the number of DCT coefficients from 4 to 7. Using six DCT basis (as chosen ad-hoc for the initial experiments) to model the pitch and energy contours seems to be adequate.

In [Kockmann et al., 2010c], the proposed approach to generate simple prosodic contour features was compared to other approaches known in the literature. The main focus was on the curve approximation technique and the segmentation technique.

Besides the two segmentation approaches presented here in Section 3.1.2, special attention was paid to two techniques found in the literature:

# of coefficients	EER [%]
4	26.11
5	25.77
6	**25.73**
7	27.29

Table 3.2: Pitch & Energy contours modeled by different numbers of DCT coefficients. NIST SRE 2006 core condition. Relevance MAP GMM-UBM system.

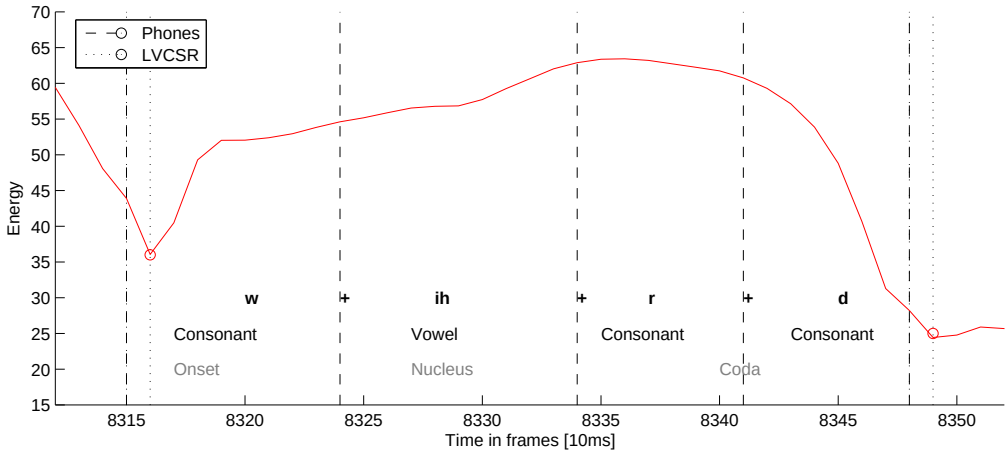


Figure 3.4: Example for different segmentations for the word weird.

1. SRI defines suprasegmental units from the output of a Large-Vocabulary Continuous Speech Recognition (LVCSR) system using a simple maximum onset algorithm (Section 3.4.1 of [Ferrer, 2009]) on the phone-level alignments. This technique is highly-complex and language-dependent but results in very accurate language-specific syllable units.

2. In [Dehak et al., 2007], it is proposed to use the technique originally described by Lin [Lin and Wang, 2005] for language identification. This approach is very simple and only needs the extracted energy values as an input. Local minima of the energy contour define the boundaries of the prosodic units.

Figure 3.4 exemplifies the investigated segmentation techniques for a snapshot of an utterance containing the word "weird". It can be observed that all three data driven techniques result in a reasonable segmentation for this example. The LVCSR segmentation (dotted lines), the pseudo-syllable segmentation derived from the phones (dashed lines) as well as the energy minima segmentation (red dots) are nearly identical.

Again, the experiments report EER on the NIST 2006 data set. However, Joint Factor

Segmentation	EER [%]
Fixed window	12.1
Energy valleys	13.7
Pseudo syllables	12.5
LVCSR syllables	**11.2**

(a) DCT.

Segmentation	EER [%]
Fixed window	12.1
Energy valleys	14.1
Pseudo syllables	12.6
LVCSR syllables	**11.4**

(b) Legendre Polynomials.

Table 3.3: Comparison of different segmentation techniques for DCT based contour approximation and Legendre polynomials. NIST SRE 2006 core condition. JFA system.

Analysis (JFA) modeling was used, as was also proposed in [Dehak et al., 2007] leading to much better results in general, compared to the previous results presented. Gender-dependent UBMs are first trained, followed by a gender-dependent JFA model with 50 eigenvoices and 20 eigenchannels trained on the same background set and scoring was based on the fast-scoring technique (see Chapter 4.3.1 for details on JFA modeling). Further, all scores are normalized using zt-norm [Auckenthaler et al., 2000].

The first experiments were carried out to compare the different segmentation techniques. Pitch and energy were modeled with six DCT coefficients using only the voiced frames. As shown in Table 3.3.a, the type of segmentation affects the EER by about 30% relative. It is interesting to see, that the complexity of the segmentation mostly corresponds to the results. The most accurate LVCSR syllables give the best EER of 11.2%, while the energy performs the worst. Surprisingly, the most simple way of fixed windows results in the second best EER of 12.1%. The results of the fixed-frame segmentation may indicate, that long time span is more crucial than correct phonetic alignment of the syllable-like units.

The following experiments show the effect of different contour modeling and further consolidate segmentation results. The setup is kept, only the curve fitting algorithm is switched from DCT to Legendre polynomials as it is proposed in [Dehak et al., 2007]. The results in Table 3.3.b show the same trend, the best EER is achieved for LVCSR with 11.4%, nearly the worst for energy with 14.1%, while the DCT modeling generally leads to little lower error-rates.

In later experiments during the system development for NIST SRE 2010, it was found that reducing the frame-shift of the fixed sized windows (so increasing the overlap and number of extracted feature vectors) further decreased the error rate and significantly outperformed more complex segmentation methods [Brümmer et al., 2010], leading to an EER of under 10%.

Processing of unvoiced	EER [%]
Voiced frames only for f0 and energy	11.4
Voiced f0 range, keep gaps, same frames energy	11.1
Voiced f0 range, keep gaps, all energy	**11.0**
Interpolation of f0, all frames f0 & energy	11.7

Table 3.4: Different processing of unvoiced regions. NIST SRE 2006 core condition. JFA system.

Dealing with undefined values

In the published work [Kockmann et al., 2010c], in addition to the curve-fitting algorithm itself, processing of undefined values (no pitch) was explored with the LVCSR segmentation setup using Legendre polynomials from Table 3.3.b. Four possible ways are compared:

1. Using only voiced frames for pitch and energy. This way, all frames where no pitch is detected are cut out prior to the curve-fitting. Note, that this reduces the length of the segment as it collapses the frames.

2. Using pitch from the first to the last voiced frame in the segment (as before), but keeping possible gaps in the pitch trajectory. This means that the supporting points to compute the Legendre polynomials are only defined at voiced frames. Although energy values are always defined, only the same frames are used to compute the energy polynomials.

3. The same frames for pitch as in 2., but using all energy in the segment (from first to last frame).

4. Linear interpolation of pitch, so all frames are used for pitch and energy, as there are no non-defined pitch values after the interpolation.

In Table 3.4, generally better results are achieved when the contour is modeled over the gaps, which suggests that preserving the pitch trajectory structure is important. The best result of 11% is achieved with the third method, so even the use of energy in unvoiced regions improves the modeling. However, the improvement is not significant. Interpolation of pitch in unvoiced regions seems to harm rather than help, mainly due to many segments that result in a straight line for pitch.

3.2 Syllable-based NERFs (SNERFs)

Besides the simple prosodic contour features presented in the previous section, a goal of the thesis was also to investigate into the use of other prosodic features, that have been

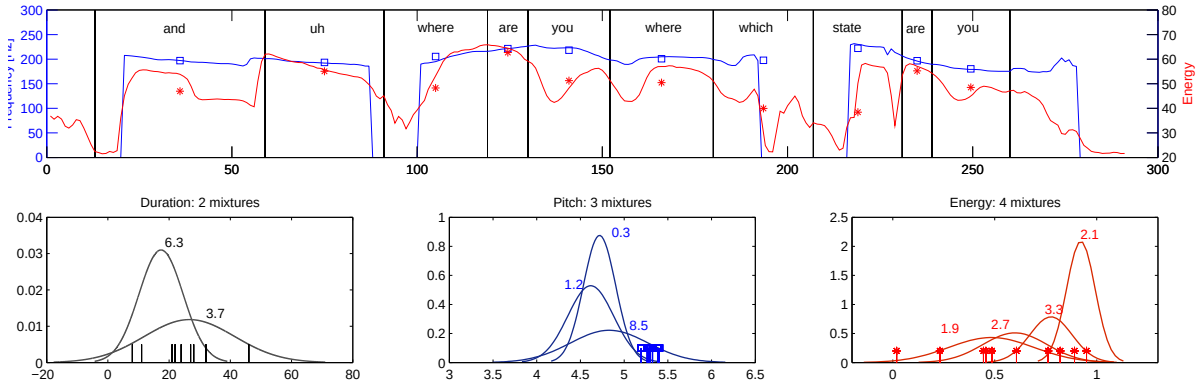


Figure 3.5: **Top row:** Extraction of three SNERF parameters from a speech segment containing 10 single-syllable words: Syllable duration (determined by black vertical lines), mean pitch value per syllable (blue squares), and mean energy per syllable (red stars). **Lower row:** Parameterization of SNERF sequences: Small GMMs are trained on background data for each individual SNERF. Two mixtures are used for duration, three mixtures for pitch, and four mixtures for energy. Occupation counts for the values extracted in the top row (here shown as bars) are collected using the GMMs.

proposed in [Shriberg et al., 2005] and successfully used in diverse systems for prosodic speaker verification [Ferrer et al., 2007, Kajarekar, 2009]. SNERFs [Shriberg et al., 2005] are Syllable-based, Non-uniform Extraction Region Features based on F0, energy, and duration information. A brief description of SNERFs follows, for details, the reader is referred to [Shriberg et al., 2005, Ferrer, 2009]. These features are actually a super-set of the contour features presented in the last section. The motivation behind SNERFs is to compute a very large number of highly correlated (differing, for example, only in the way they are normalized) prosodic features in order to let the modeling approach choose the most important information. On contrary to simple contour features, SNERFs also model pauses and keep feature frames with undefined values, resulting in a changing number of undefined measurements. Further, they are of much higher dimensionality and include both continuous and discrete measurements. For this reason, the modeling techniques used for these features differ significantly from the JFA based approaches used in the previous experiments.

These features were used as provided by SRI without any modification[1] to develop an appropriate subspace modeling technique in Chapter 4.4.

3.2.1 Basic SNERFs

Duration features

Segmentation and duration information is also based on syllable units, but the used syllable segmentation is generated from the output of a Large-Vocabulary Continuous Speech Recognition (LVCSR) system using a simple maximum onset algorithm (Section 3.4.1 of [Ferrer, 2009]) on the phone-level alignments.

On contrary to the simple contour features proposed before, six different duration features are used from each syllable region: duration of the onset, nucleus, coda, onset plus nucleus, nucleus plus coda and the full syllable duration. Note, that except the syllable and nucleus duration, every other measurement may be undefined. Further, all measurements are used as they are and are also normalized using statistics from held-out data.

Pitch features

First, pitch features are extracted for each utterance using the same correlation based algorithm as described in Section 3.1.1. Then, pitch based SNERFs are generated for two different regions: voiced frames in the syllable and voiced frames that are not suspicious to halving or doubling (this is estimated by the log-tied-model introduced in [Sönmez et al., 1997] and mentioned in Section 1.3.2). The pitch output in these two regions is then used in raw form, as well as median filtered and stylized (as exemplified in Figure 1.7 in the introduction). Further, for each pitch sequence, a large set of features is computed: maximum, minimum, mean, maximum minus minimum, number of rising/ falling/doubled/halved or voiced frames, length of the slopes in the sequence, number of changes from rising to falling, values of the first/last or averaged slope and the maximum positive or negative slope. Again, all these measurements are used as they are and are further normalized using statistics obtained from held-out data. Legendre polynomial approximations similar to our proposed contour features are also computed, but even for different regions (syllable and nucleus) and with several number of coefficients (first, third and fifth order).

Energy features

Raw short-term energy measures are first computed from the signal. Again, different regions are used to compute energy- based SNERFs: nucleus, nucleus without unvoiced frames, the whole syllable, and the whole syllable without unvoiced frames. The segment energies are also used in raw and stylized form and are further processed similar to the pitch sequences.

[1]Many thanks for providing their features and support for building the baseline system to Luciana Ferrer and SRI International.

3.2.2 SNERFs tokens

Further, long-term dependencies are modeled by concatenating features from consecutive syllables and pauses. New vectors are formed for each basic feature by concatenating consecutive values. If a pause is found within the sequence, the length of the pause is used as a feature. For example, for trigrams, five different patterns: (S, S, S), (P, S, S), (S, P, S), (S, S, P), (P, S, P) are obtained, where P represents pause and S a syllable in the sequence of concatenated feature vectors. Each pair {feature, pattern} determines what is called a *token* (see [Ferrer et al., 2007] for details). For each token, the feature distribution is modeled independently in the next step as described in 3.2.4. The current implementation uses sequences of lengths one, two, and three, and a total of nine different patterns.

3.2.3 Final SNERFs

The first line of the plots in Figure 3.5 shows a simplified example of the feature extraction process. The segments are determined by the syllables found in the ASR output. The pitch (blue curve) and energy (red curve) signals are estimated from the waveform. For this example, it is assumed that only three features per segment are extracted: the whole unnormalized syllable duration (from one vertical black line to the next), the mean of the raw pitch value (blue squares), and the mean of the normalized energy value (red stars).

After extraction of the SNERFs, high-dimensional feature vectors are obtained per utterance. Even only for the syllable uni-gram token S, a 182-dimensional feature vector per syllable is used. Further, the values in each multidimensional feature frame consists of discrete, continuous or even undefined features. These two properties – the high dimensionality as well as the heterogeneous nature – makes it impossible to train a standard GMM for them as it is done for the simple low-dimensional and well-defined DCT contour features as described previously.

3.2.4 Parameterization of SNERFs

In [Ferrer et al., 2007] several techniques are proposed to parameterize the sequences of SNERF frames to a fixed-length representation per utterance, which can in turn be used as input to various classifiers like SVMs. The basic idea in all approaches is to divide the dynamic range of each single SNERF value into discrete bins and to count how often the values of an utterance fall into a certain bin. The best approach is based on using soft bins defined by components of GMMs.

For each token, a separate Gaussian Mixture Model is trained with a small number of mixture components on the background data. Because basic features may be undefined (e.g., when no pitch is detected or when the syllable lacks onset or coda), a special GMM is needed using additional parameters for the probability of a feature being undefined. In the first pass, all GMMs are trained using frames with defined features only and the model is trained as a standard GMM. The GMMs are then retrained with all feature vectors, training

also the additional parameter (a prior whether the feature is defined for the particular Gaussian). When computing the likelihood of the data using the modified algorithm, the standard Gaussian likelihood is simply multiplied by its prior of being defined, if the feature is defined. Otherwise, solely the prior determines the likelihood. Re-estimation formulae of the modified expectation-maximization algorithm are given in [Kajarekar et al., 2004].

The second line of Figure 3.5 shows a toy example in which three small GMMs are trained on a background data set. A two-component model is trained for the syllable durations, a three-component model for mean pitch values, and a four-component GMM for means of syllable energies.

After training the background models for each token, Gaussian component occupation counts are gathered for each utterance (zero order sufficient statistics from the modified EM algorithm [Kajarekar et al., 2004]). These are accumulated soft counts describing the responsibilities of each individual mixture component toward generating the frames in the utterance. Using these parameters, the sequence of SNERFs (one feature vector per syllable) is transformed to fixed length vectors (one vector of soft counts per utterance).

The values from the exemplified feature extraction process (syllable duration, mean pitch, and mean energy) are further depicted as bars in the second row of Figure 3.5. The occupation counts (the numbers next to the mixtures) are the responsibilities for each Gaussian component in generating these values. Each Gaussian component can be seen as a discrete class and the occupation counts can be seen as soft-counts of discrete events.

The described parameterization process for SNERFs will be used through the thesis and an appropriate subspace model for these counts will be developed in Section 4.4.

4

Modeling approaches for prosodic speaker verification

> The problem of finding and exploiting low-dimensional structures in high-dimensional data is taking on increasing importance in image, video, or audio processing; Web data analysis/search; and bioinformatics, where data sets now routinely lie in observational spaces of thousands, millions, or even billions of dimensions. The curse of dimensionality is in full play here: We often need to conduct meaningful inference with a limited number of samples in a very high-dimensional space. Conventional statistical and computational tools have become severely inadequate for processing and analyzing such high-dimensional data [Ma et al., 2011].

Dimensionality reduction via subspace learning is of great interest in the whole signal processing community and we also tackle the same problems in speaker verification. Take a standard speaker verification system with 60-dimensional MFCC based features that are modeled using 2048-component GMMs. More than 120,000 mean parameters have to be estimated per model often with only about one thousand feature vector instances (for a signal of 10 seconds).

While adaptation from a properly estimated background model [Reynolds et al., 2000] greatly improves the robustness of speaker models, several techniques have been developed to decrease the number of parameters that have to be estimated while keeping the discriminative power of the model.

Deterministic techniques like Principal Component Analysis (PCA) [Jolliffe, 2002] or Linear Discriminant Analysis (LDA) [Fukunaga, 1990] are often used to directly reduce the feature dimensionality in the front-end processing. However, in the last years, probabilistic continuous latent variable models, similar to the Factor Analysis (FA) model [Bishop, 2006, Chapter 12], have been proposed to efficiently model the intrinsic complexity in a low-dimensional model parameter space.

In this chapter, the basic GMM approach to speaker verification will be introduced in more detail first, as a quick introduction to GMMs was already given in Section 1.3.1. Next, subspace modeling techniques for Gaussian mean parameters are described as used in cepstral baseline systems and also for the proposed contour features presented in Section

3.1. In Section 4.4, a novel subspace model for multinomial distributions is proposed, that is applicable to the parameterized SNERFs as described in Section 3.2. Generally, only the formulae needed to implement the algorithms are provided. Detailed derivations for the subspace models can be found in Appendices A and B.

4.1 Standard UBM-GMM with MAP adaptation

Many speaker recognition models used nowadays are based on a standard Universal Background Model-Gaussian Mixture Modeling (UBM-GMM) [Reynolds et al., 2000] paradigm. In this section, the focus is on re-estimation of the model parameters and the likelihood evaluation. All GMMs used in our experiments are multivariate with dimension D and contain C Gaussian mixture components.

Prior to any other model training, a speaker-independent model is trained on pooled feature vectors $\mathbf{O} = [\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3, \ldots, \mathbf{o}_n]$ $(D \times N)$ of all N frames in the background data. It is called Universal Background Model. Weights π_c, means $\boldsymbol{\mu}_c$ and variances $\boldsymbol{\Sigma}_c$ of each UBM component c are trained in a maximum-likelihood way with an Expectation-Maximization (EM) algorithm. For a GMM, EM [Dempster et al., 1977] iteratively alternates between estimating the responsibilities $\gamma_k(n)$ (E-Step, posterior probability of Gaussian component $c = 1 \ldots C$ generating frame $n = 1 \ldots N$) and re-estimation of the model parameters using the current responsibilities (M-Step).

E-Step:

$$\gamma_c(n) = \frac{\pi_c \mathcal{N}(\mathbf{o}_n | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}{\sum_{j=1}^{C} \pi_c \mathcal{N}(\mathbf{o}_n | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}. \tag{4.1}$$

For an efficient batch processing of the M-Step, sufficient statistics are defined that are accumulated over all the training data:

$$\text{Zero Order:} \quad \gamma_c = \sum_{n=1}^{N} \gamma_c(n). \tag{4.2}$$

$$\text{First Order:} \quad \boldsymbol{\theta}_c = \sum_{n=1}^{N} \gamma_c(n) \mathbf{o}_n. \tag{4.3}$$

$$\text{Second Order:} \quad \boldsymbol{\Theta}_c = \sum_{n=1}^{N} \gamma_c(n) \mathbf{o}_n \mathbf{o}_n^T. \tag{4.4}$$

M-Step: The mean vectors can be updated as

$$\boldsymbol{\mu}_c^{new} = \frac{1}{\gamma_c} \boldsymbol{\theta}_c \tag{4.5}$$

and the covariance matrices can be efficiently updated using

$$\boldsymbol{\Sigma}_c^{new} = \frac{1}{\gamma_c}\boldsymbol{\Theta}_c - \boldsymbol{\mu}_c^{new}\boldsymbol{\mu}_c^{newT}. \tag{4.6}$$

Note, that only GMMs with diagonal covariances are used in this work. The mixture weights are updates as:

$$\pi_c^{new} = \frac{\gamma_c}{N}. \tag{4.7}$$

The likelihood function for each component is

$$\mathcal{N}(\mathbf{o}_n|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}_c|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{o}_n - \boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\mathbf{o}_n - \boldsymbol{\mu}_c)\right\} \tag{4.8}$$

for feature vector $\mathbf{o}_n$ with feature dimension D. The data log-likelihood for the whole GMM and all data $\mathbf{O}$ is given as

$$\ln p(\mathbf{O}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \pi) = \sum_{n=1}^{N}\ln\sum_{c=1}^{C}\pi_k\mathcal{N}(\mathbf{o}_n|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c). \tag{4.9}$$

For parameter updates using the EM algorithm, the likelihood of the training data increases at each iteration. This can be checked for convergence during the training.

Following the UBM training, individual speaker models can be obtained by relevance Maximum-A-Posteriori (MAP) adaptation [Reynolds et al., 2000] of the mean parameters using the enrollment data only. Weights and variances are kept fix.

For a better understanding of the MAP adaptation, lets first assume the more general case of Bayesian inference of the mean parameter with known variance. For simplicity, lets look at a single Gaussian distributed random variable x. The posterior distribution of the mean parameter μ given N data points in $\mathbf{x}$ is [Bishop, 2006, Chapter 2]:

$$p(\mu|\mathbf{x}) \propto p(\mathbf{x}|\mu)p(\mu) \tag{4.10}$$

with the likelihood function $p(\mathbf{x}|\mu) = \mathcal{N}(\mathbf{x}|\mu, \sigma^2)$. If we assume a Gaussian prior for the mean parameter

$$p(\mu) = \mathcal{N}(\mu|\mu_0, \sigma_0^2), \tag{4.11}$$

the posterior will also be Gaussian and after some manipulation we will find the posterior to be:

$$p(\mu|\mathbf{x}) = \mathcal{N}(\mu|\mu_N, \sigma_N^2). \tag{4.12}$$

For MAP adaptation we are only interested in the point estimate of μ, for which the posterior is maximum (i.e. the most probable value of μ). In the case of Gaussians, it is the mean μ_N of the distribution, and can be written as:

$$\mu_N = \frac{\sigma^2}{N\sigma_0^2 + \sigma^2}\mu_0 + \frac{N\sigma_0^2}{N\sigma_0^2 + \sigma^2}\mu_{ML} \tag{4.13}$$

with μ_{ML} being the maximum likelihood solution for μ.

Now, from this general solution for the MAP point estimate of a Gaussian mean parameter we can draw the link to the relevance MAP adaptation formula usually used in speaker verification [Reynolds et al., 2000], that is often presented and understood as a *rule of thumb* of just interpolating between the UBM mean and the maximum likelihood solution.

In relevance MAP adaptation, we set the mean μ_0 of the prior $p(\mu)$ to be the UBM mean and the the variance of the prior σ_0^2 is ad-hoc set to be a fraction of the diagonal UBM variance:

$$\sigma_0^2 = \frac{\sigma^2}{\tau}. \tag{4.14}$$

Inserting this into 4.13, the MAP estimate becomes

$$\mu_N = \frac{1}{N/\tau + 1}\mu_0 + \frac{N}{N+\tau}\mu_{ML}. \tag{4.15}$$

Generalizing this to a multivariate Gaussian mixture model and assuming prior distributions for individual Gaussians and the individual coefficients to be statistically independent, we can write the prior distribution of the supervector of GMM means as:

$$p(\mathbf{\Phi_s}) = \mathcal{N}(\mathbf{\Phi_s}|\mathbf{m}, \mathbf{D}^2), \tag{4.16}$$

where the mean supervector $\mathbf{\Phi}$ is constructed by stacking all the mean vectors $\boldsymbol{\mu}_c$of all the Gaussian components into a single large dimensional supervector $\mathbf{\Phi}_s = [\mu_{s1}^T, \mu_{s2}^T, \mu_{s3}^T, \ldots, \mu_{sC}^T]^T$. The CD dimensional mean $\mathbf{m}$ contains the stacked mean vectors of the UBM. Further, the C block-diagonal elements of the $CD \times CD$ dimensional matrix $\mathbf{D}$ are set to be $\sqrt{\mathbf{\Sigma}_c/\tau}$, with $\mathbf{\Sigma}_c$ being the diagonal UBM covariance of component c.

For each component of the supervector, we obtain the well-know update equation for the mean of an enrolled speaker model:

$$\boldsymbol{\mu}_{sc}^{MAP} = \alpha_c \boldsymbol{\mu}_{sc}^{ML} + (1-\alpha_c)\boldsymbol{\mu}_c^{UBM} \tag{4.17}$$

with adaptation coefficients

$$\alpha_c = \frac{\sum_{n=1}^{N}\gamma_c(n)}{\sum_{n=1}^{N}\gamma_c(n) + \tau}, \tag{4.18}$$

and relevance factor τ (usually 4–16).

During verification, scores are obtained as the Log-Likelihood Ratio (LLR) between the speaker model- and the UBM log-likelihood for the test utterance u, evaluating Equation 4.9 for both, the speaker model and the UBM:

$$LLR = \ln p(\mathbf{O}_u|\boldsymbol{\mu}_s, \mathbf{\Sigma}_s, \pi_s) - \ln p(\mathbf{O}_u|\boldsymbol{\mu}_{UBM}, \mathbf{\Sigma}_{UBM}, \pi_{UBM}). \tag{4.19}$$

For computational efficiency, only top scoring Gaussians (determined based on the UBM) are evaluated for the speaker models per frame.

4.2 Introducing Joint Factor Analysis models

In [Bishop, 2006, Chapter 12] Factor Analysis is described as a linear-Gaussian latent variable model, where the conditional distribution of an observed variable $\mathbf{x}$ given an latent variable $\mathbf{z}$ can be written as:

$$\mathbf{x} = \mathbf{W}\mathbf{y} + \mathbf{D}\mathbf{z}, \tag{4.20}$$

with $\mathbf{D}$ being a diagonal covariance matrix and $\mathbf{z}$ being a standard normal distributed Gaussian variable. Generally, the factor analysis model *factors* the observed covariance structure of the data. The independent variance for each variable in $\mathbf{x}$ is covered in $\mathbf{D}$, while the correlations between the variables are captured in the matrix $\mathbf{W}$. The marginal distribution of the observed variable is given by $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$ with

$$\mathbf{C} = \mathbf{W}\mathbf{W}^T + \mathbf{D}^2. \tag{4.21}$$

Joint Factor Analysis [Kenny and Dumouchel, 2004] as introduced in GMM based speaker verification differs from the above model in some aspects. First of all, the FA model in speaker verification does not model the distribution of features (such as MFCCs), but models the prior distribution of speaker dependent mean supervectors of a GMM $\boldsymbol{\Phi}_s$, which are assumed to be Gaussian distributed. With this in mind, lets try to formulate the classical relevance MAP adaptation as a FA model similar to 4.20. In relevance MAP, the covariances $\boldsymbol{\Sigma}_c$ for each component are only diagonal, so they do not account for the correlations between different Gaussians or even feature dimensions. The conditional distribution of a speaker dependent mean supervector $\boldsymbol{\Phi}_\mathbf{s}$ can be written as:

$$\boldsymbol{\Phi}_\mathbf{s} = \mathbf{m} + \mathbf{D}\mathbf{z}_s, \tag{4.22}$$

with $\mathbf{m}$ and $\mathbf{D}$ already defined for Equation 4.16 and $\mathbf{z}$ being the CD dimensional latent variable.

As it is the intention of the FA model, it is much more reasonable to let the factor loading matrix capture correlations between observed variables. In the case of a GMM supervector we are even able to learn correlations between the individual components. In the case of speaker verification, we should be able to learn the correlations of the observed mean supervectors belonging to different speakers. This way, we could learn a more-reasonable prior covariance of the supervector mean distribution. Further, its is reasonable to assume that most of the variability between speakers can already be captured in a low-dimensional subspace, which allows us to use a low-dimensional factor loading matrix $\mathbf{V}$ ($CD \times R$, with $R << CD$). This is the main idea of eigenvoice MAP [Kenny et al., 2003]:

$$\boldsymbol{\Phi}_s = \mathbf{m} + \mathbf{V}\mathbf{y}_s. \tag{4.23}$$

Here, all variability in the model is assumed to be covered by the low-dimensional matrix $\mathbf{V}$. As the dimension of $\mathbf{y}_s$ is much smaller than $\mathbf{z}_s$, eigenvoice MAP tends to converge with much less adaptation data than classical MAP.

Further, [Kenny et al., 2003] proposed a similar model to capture channel- in addition to speaker differences. The so called eigenchannel model assumes that a low-dimensional factor loading matrix $\mathbf{U}$ corresponds to a low-dimensional subspace, with $\mathbf{U}\mathbf{U}^T$ being the non-full rank covariance representing largest within speaker variability in the prior distribution of the mean supervector:

$$\mathbf{\Phi}_{su} = \mathbf{\Phi}_s + \mathbf{U}\mathbf{x}_{su}. \tag{4.24}$$

Here, $\mathbf{\Phi}_s$ is already a speaker adapted mean supervector (either by classical or eigenvoice MAP) and the within speaker variability in the mean parameters is modeled using the subspace $\mathbf{U}$ based on the utterance dependent posterior distribution of $\mathbf{x}_{su}$.

All these FA models are based on the same mathematical framework and differ only in the way the factor loading matrices are restricted and estimated.

Finally, [Kenny, 2006] proposed an integrated *Joint Factor Analysis* approach combining the three ways of factoring the speaker- and utterance dependent correlations.

$$\mathbf{\Phi}_{su} = \mathbf{m} + \mathbf{V}\mathbf{y}_s + \mathbf{U}\mathbf{x}_u + \mathbf{D}\mathbf{z}_s. \tag{4.25}$$

This way, channel effects could be considered during speaker adaptation and vice versa by considering joint distribution of all latent variables, making this model most efficient. The next Section will describe the model in more detail, including the assumptions and approximations we make to justify our re-estimation framework.

4.3 Subspace models for parameters of Gaussian distributions

4.3.1 Separate speaker and channel subspaces

The classical formulation of JFA for speaker verification [Kenny, 2006, Kenny et al., 2008b] assumes that the concatenated speaker and utterance specific mean vectors $\boldsymbol{\phi}_{\text{GaussJFA}} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3, \ldots, \boldsymbol{\mu}_C]$ of a Gaussian mixture model are distributed according to a subspace model with separate subspaces for speaker and channel variability and a residual speaker variability:

$$\boldsymbol{\phi}_{\text{GaussJFA}} = \mathbf{m} + \mathbf{V}\mathbf{y}_s + \mathbf{U}\mathbf{x}_u + \mathbf{D}\mathbf{z}_s, \tag{4.26}$$

where $\mathbf{m}$ is a DC dimensional speaker- and channel-independent super-vector, usually built from the concatenated UBM mean vectors. $\mathbf{V}$ and $\mathbf{U}$ span linear low-dimensional subspaces (for speaker and channel variability) in the original mean supervector parameter space. The subspace dimensionalities $R_{\mathbf{V}}$ and $R_{\mathbf{U}}$ are much smaller than DC, typically between 50 and 300. $\mathbf{D}$ is a diagonal matrix of full rank. The components of $\mathbf{y}$ and $\mathbf{x}$ are the low-dimensional latent variables corresponding to the speaker and channel subspaces. $\mathbf{z}$ are latent variables corresponding to the diagonal matrix of residual speaker variability not covered by the subspace $\mathbf{V}$. However, the $\mathbf{D}\mathbf{z}$ term has shown to be redundant in terms of performance [Burget et al., 2009b] and it will not be considered further.

The key idea of JFA is to learn low-dimensional linear subspaces ($\mathbf{V}$ and $\mathbf{U}$) in which the model parameters live. During speaker model training, only the low-dimensional latent variable vectors $\mathbf{y}$ and $\mathbf{x}$ are estimated jointly, which makes the speaker model estimation very robust on small amounts of data, as the model is restricted to move inside the subspaces. During verification, only the low-dimensional latent variable vector $\mathbf{x}$ is estimated for the test utterance, allowing the model parameters to further adapt towards the channel condition in the test utterance along the channel subspace $\mathbf{U}$. See also Figure 1.5 in the introduction.

The used JFA framework makes some assumptions to make model estimation practical on large databases:

- The alignment of feature frames to Gaussian components is solely based on the UBM. This way, the sufficient statistics used in the re-estimation only have to be collected once per utterance.
- Joint estimation of speaker and channel variability for several recordings stemming from the same speaker is computational very demanding, so a simplified training procedure, assuming decoupled speaker and channel effects, is used [Kenny et al., 2008b].
- Our hyperparameter matrices $\mathbf{U}$ and $\mathbf{V}$ are set to specific values after each iteration. This framework is known as type 2 maximum likelihood or evidence approximation (see [Bishop, 2006, Chapter 3]).

Note, that only the used framework is described and that it also differs in many aspects (e.g. simplifications in the likelihood calculations) from the originally proposed JFA framework by Kenny [Kenny et al., 2005]. We use the following scheme (see also Algorithm 1):

1. Iterative training of the speaker variability subspace $\mathbf{V}$ on a large database ($\mathbf{U}$ is zero).
2. Iterative training of the channel variability subspace $\mathbf{U}$ on a large database (with fixed $\mathbf{V}$).
3. Enrollment of each speaker by point MAP estimates of the latent variables for the enrollment data.
4. Channel point estimate for each test utterance (based on UBM).
5. Likelihood calculation based on MAP point estimates.

During the iterative procedure of re-estimating $\mathbf{V}$ and then $\mathbf{U}$, we always need to estimate the posterior distribution of the corresponding latent variables using the current model parameters. The JFA model is a probabilistic model where the latent variables $\mathbf{y}$ and $\mathbf{x}$ are distributed according to a zero-mean unit-covariance Gaussian prior:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{I}), \quad p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{I}). \tag{4.27}$$

With the specific form of the JFA likelihood function (using MAP point estimates of the model parameters) and the Gaussian prior, it is possible to obtain the posterior distribution of the latent variables using a closed form solution for each speaker s and utterance u in the training data:

$$p(\mathbf{y}_s|\mathbf{O}_s) = \mathcal{N}(\mathbf{y}_s|\hat{\mathbf{y}}_s, \mathbf{L}_s^{-1}), \quad p(\mathbf{x}_u|\mathbf{O}_u) = \mathcal{N}(\mathbf{x}_u|\hat{\mathbf{x}}_u, \mathbf{L}_u^{-1}) \tag{4.28}$$

During hyperparameter estimation, the speaker factors are computed using all utterances of a speaker $\mathbf{O}_s$ and the corresponding fixed channel factors (which are zero in our case). For the speaker factors $\mathbf{y}_s$ the MAP estimate is given as:

$$\hat{\mathbf{y}}_s^T = \sum_c \sum_u (\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\hat{\mathbf{x}}_u^T\mathbf{U}_c^T)\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{L}_s^{-1}, \tag{4.29}$$

where $\mathbf{V}_c$ is the $D \times R_\mathbf{V}$ block of $\mathbf{V}$ corresponding to the cth component, with statistics γ_{uc} and $\boldsymbol{\theta}_{uc}$ collected for each utterance u of speaker s as defined by Equations 4.2 and 4.3. $\boldsymbol{\Sigma}_c$ is the $D \times D$ UBM covariance matrix and the precision matrix of the posterior distribution is

$$\mathbf{L}_s = \sum_c \sum_u \gamma_{uc}\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c + \mathbf{I}_\mathbf{V}, \tag{4.30}$$

where $\mathbf{I}_\mathbf{V}$ is an $R_\mathbf{V} \times R_\mathbf{V}$ identity matrix.

Using a large database with many utterances from many speakers and keeping the distributions of the latent variables for each utterance fixed, it is in turn possible to reestimate the subspace matrix $\mathbf{V}$.

The submatrix $\mathbf{V}_c$ corresponding to cth block of $\mathbf{V}$ can be re-estimated in the ML II framework as

$$\mathbf{V}_c = \sum_s \sum_u [(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{U}_c\hat{\mathbf{x}}_u)\hat{\mathbf{y}}_s^T(\hat{\mathbf{y}}_s\hat{\mathbf{y}}_s^T + \mathbf{L}_s^{-1}\gamma_{uc})^{-1}]. \tag{4.31}$$

After several iterations the subspace $\mathbf{V}$ is fixed and the channel subspace $\mathbf{U}$ can be estimated. In this process, the channel factors are computed per utterance $\mathbf{O}_u$ using the fixed subspace $\mathbf{V}$ and the fixed speaker factors for the corresponding speaker.

Analogously, for the channel factors $\mathbf{x}$, the mean of the posterior distribution is given as:

$$\hat{\mathbf{x}}_u^T = \sum_c (\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\hat{\mathbf{y}}_{s\leftarrow u}^T\mathbf{V}_c^T)\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{L}_u^{-1}, \tag{4.32}$$

where $\mathbf{U}_c$ is the $D \times R_\mathbf{U}$ block of $\mathbf{U}$ corresponding to the cth component, with $\hat{\mathbf{y}}_{s\leftarrow u}$ being the speaker factors corresponding to the speaker of utterance u. The precision matrix of the posterior distribution is defined as

$$\mathbf{L}_u = \sum_c \gamma_{uc}\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c + \mathbf{I}_\mathbf{U}, \tag{4.33}$$

with $\mathbf{I}_\mathbf{U}$ being an $R_\mathbf{U} \times R_\mathbf{U}$ identity matrix.

Again, an analogous solution can be obtained for the channel subspace $\mathbf{U}_c$

$$\mathbf{U}_c = \sum_s \sum_u [(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{V}_c\hat{\mathbf{y}}_s)\mathbf{x}_{\hat{u}}^T(\hat{\mathbf{x}}_u\hat{\mathbf{x}}_u^T + \mathbf{L}_u^{-1}\gamma_{uc})^{-1}]. \quad (4.34)$$

The full derivations and update formulae for the JFA framework as used through this thesis can be found in Appendix A.

Algorithm 1 Quasi algorithm of used JFA modeling framework

```
U, V ← random
{Estimate speaker subspace}
x_train ← 0
for i = 1 → nIt do
  for all speaker in background data do
    Compute posterior distribution of y per speaker (Equations 4.29 and 4.30)
  end for
  ML II update of V (Equation 4.31)
end for
{Estimate channel subspace}
Keep MAP estimates of y per speaker
for i = 1 → nIt do
  for all utterances in background data do
    Compute posterior distribution of x per utterance (Equation 4.32 and 4.33)
  end for
  ML II update of U (Equation 4.34)
end for
{Enroll speaker model}
for all speaker in training data do
  Compute MAP estimate of y and x jointly per speaker (Equation 4.29)
end for
{Verify}
y_test ← 0
for all utterances in test data do
  Compute MAP estimate of x per utterance (Equation 4.32)
  for all speaker in training data do
    Compute LLR (Equation 4.35)
  end for
end for
```

To enroll a speaker using a single enrollment utterance, the MAP point estimates of the speaker and channel factors are computed jointly using 4.29 (by simply stacking $\mathbf{V}$ and $\mathbf{U}$ in a single subspace matrix). This way channel effects are also taken into consideration during enrollment. However, only $\mathbf{y}$ is kept afterwards.

By using solely point estimates in the enrollment and verification process, the speaker- and channel adapted mean supervector can be constructed using the JFA model for each *enrollment-speaker – test-utterance* combination. Here, we make another assumption:

- The channel factors during verification are solely estimated using the UBM ($\mathbf{y}$ set to $\mathbf{0}$). This way, the channel has to be estimated only once per test utterance, instead of estimating it for each test-utterance/model combination in test.

Experiments indicated that this simplification does not harm the performance but greatly reduces the computational load.

Then, the same frame-by-frame Log Likelihood Ratio (LLR) computation (Equation 4.19) can be used for scoring as for the standard approaches without subspace modeling. However, more efficient approaches have been proposed and were compared by [Glembek et al., 2009].

The so-called *fast scoring* technique for JFA has proven to be much more computationally efficient than the standard likelihood evaluation. Fast scoring for JFA is based on further approximations:

- The assignment of feature frames to Gaussian components is solely based on the UBM. The scoring technique uses the same sufficient statistics as are used in UBM or JFA parameter re-estimation (Equations 4.2 and 4.3). This way, the GMM evaluation of each feature frame is only needed once, using the UBM, and further processing steps are only based on the fixed-sized statistic vectors.

- As the statistics based on the UBM are used, the true likelihood of the data given the model is approximated by the same auxiliary function as used in an EM algorithm (see also Section A.1).

- The quadratic auxiliary function is further simplified by a linear approximation using first-order Taylor series.

This reduces the computational demand to a simple dot-product scoring of channel compensated statistic supervectors leading to the following scoring function:

$$LLR_{fast} = \sum_c \mathbf{y}_s^T \mathbf{V}_c^T \mathbf{\Sigma}_c^{-1} (\boldsymbol{\theta}_c - \gamma_c \mathbf{m}_c - \gamma_c \mathbf{U}_c \mathbf{x}). \tag{4.35}$$

The whole process of hyperparameter estimation, training and test is sketched in the quasi algorithm 1. A tutorial-style implementation of the described JFA framework can be found at [Glembek, 2009]. This framework dominated the 2008 NIST SREs and was broadly adopted during the NIST SRE 2010.

4.3.2 Total variability subspace

Besides the great success of the JFA model, it was found during the *JHU Summer Workshop on Robust Speaker Recognition* [Burget et al., 2008], that even the channel-related latent variables $\mathbf{x}$ still contain information to discriminate between speakers.

Based on these findings, [Dehak et al., 2009b] proposed a simplified variant of the JFA framework, that assumes that speaker and channel subspaces are not independent and uses only one subspace covering the total variability in an utterance:

$$\boldsymbol{\phi}_{\text{GaussIV}} = \mathbf{m} + \mathbf{T}\mathbf{w}. \tag{4.36}$$

Again, $\mathbf{T}$ spans a low-dimensional linear subspace in the original mean parameter space. Also, the subspace size $R_{\mathbf{T}}$ is much smaller than $D \times C$, usually around 400. The components of $\mathbf{w}$ are the low-dimensional latent variables corresponding to coordinates in the total variability subspace.

The key idea of the total variability modeling is to learn a single low-dimensional subspace $\mathbf{T}$ in which the model parameters live. The model can then be used to extract low-dimensional representations of each utterance (the latent factors $\mathbf{w}$), by adapting the model to each utterance. This way, the features per utterance are reduced to a single vector with a few hundred dimensions, while keeping most of the important information. This makes it possible to apply various machine learning techniques for the final verification, which were not applicable to high-dimensional sequential data.

The simplified JFA model is again a probabilistic model, where the latent variables $\mathbf{w}$ are distributed according to a standard normal prior:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I}). \tag{4.37}$$

This again leads to a closed-form solution of the posterior distribution of the latent variable:

$$p(\mathbf{w}|\mathbf{o}_u) = \mathcal{N}(\mathbf{w}_u|\hat{\mathbf{w}}_u, \mathbf{L}_u^{-1}). \tag{4.38}$$

The re-estimation of the model parameters is based on the same EM algorithm as used in the previous section. For the total variability factors $\mathbf{w}$, the mean of the posterior distribution is given as:

$$\hat{\mathbf{w}}_u^T = \sum_c (\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T)\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{T}_c\mathbf{L}_u^{-1}, \tag{4.39}$$

where $\mathbf{T}_c$ is the $D \times R_{\mathbf{T}}$ block of $\mathbf{T}$ corresponding to the cth component, with the precision matrix of the posterior distribution

$$\mathbf{L}_u = \sum_c \gamma_{uc}\mathbf{T}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{T}_c + \mathbf{I}_{\mathbf{T}} \tag{4.40}$$

and $\mathbf{I}_{\mathbf{T}}$ being an $R_{\mathbf{T}} \times R_{\mathbf{T}}$ identity matrix. Analogously to the JFA framework, keeping the distribution of the latent variables for each utterance fixed, it is in turn possible to reestimate the subspace matrix $\mathbf{T}$:

Submatrix $\mathbf{T}_c$ corresponding to cth block of $\mathbf{T}$ can be re-estimated as

$$\mathbf{T}_c = \sum_u [(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c)\mathbf{w}_u^T(\mathbf{w}_u\mathbf{w}_u^T + \mathbf{L}_u^{-1}\gamma_{uc})^{-1}]. \tag{4.41}$$

Note, that this time no speaker labels are used at all and the latent vector distributions are computed for each utterance in the training data.

In this approach, the JFA-like model serves only as the extractor of the vectors $\mathbf{w}$, which can be seen as low-dimensional fixed-sized representations of utterances, and which are in turn used as inputs to another classifier. The low-dimensional vector $\mathbf{w}$ is also known as an iVector.

Note, that unlike the standard JFA, where two subspaces are used to account for speaker and intersession variability, the iVector variant uses a single subspace accounting for all the variability. Therefore, the extracted vectors $\mathbf{w}$ are not free of channel effects, and intersession compensation must be eventually considered during classification.

[Dehak et al., 2009b] originally proposed to classify the iVectors using SVMs. Due to the low-dimensionality (usually in the order of hundreds), intersession variability compensation could be efficiently incorporated into the SVM kernel. Eventually, a simple cosine distance scoring approach that is solely measuring the angle between two iVectors resulted in the most efficient method:

$$score_{\texttt{cosine}} = \frac{\langle \mathbf{w}_1, \mathbf{w}_2 \rangle}{\|\mathbf{w}_1\| \|\mathbf{w}_2\|}. \tag{4.42}$$

For evaluating trials, its should be mentioned that using the iVector model completely symmetrizes the problem: It does not matter anymore which utterance in a trial is the training, and which is the test utterance. The task is simply to compare the two iVectors.

Normalization techniques, like zt-norm can be included very efficiently, by simply "adding" iVectors for each t-norm utterance to the training iVectors and "adding" iVectors for each z-norm utterance to our test iVectors. This way, the Gram matrix containing all cosine-distance scores already includes all scores needed to compute the zt-norm statistics.

Intersession compensation was incorporated in the iVector feature domain by standard Linear Discriminant Analysis (LDA) followed by Within-Class Covariance Normalization (WCCN) [Hatch et al., 2006]. First, the within-class and the between-class covariance matrices need to be trained on iVectors estimated on a large hold-out set. Usually, the same background set as used for the UBM or the iVector training. The between class $\mathbf{B}$ is computed as

$$\mathbf{B} = \sum_{s=1}^{S} N_s (\overline{\mathbf{w}}_s - \overline{\mathbf{w}})(\overline{\mathbf{w}}_s - \overline{\mathbf{w}})^T \tag{4.43}$$

with $\overline{\mathbf{w}}_s = \frac{1}{U_s} \sum_u^{U_s} \mathbf{w}_u$ being the mean of the iVectors per speaker and $\overline{\mathbf{w}} = \frac{1}{U} \sum_u^{U} \mathbf{w}_u$ being the mean of all iVectors in the data set. N_s is the number of iVectors per speaker s. The within class $\mathbf{W}$ is computed as

$$\mathbf{W} = \sum_{s=1}^{S} \sum_{u}^{U_s} (\mathbf{w}_u - \overline{\mathbf{w}}_s)(\mathbf{w}_u - \overline{\mathbf{w}}_s)^T. \tag{4.44}$$

The LDA matrix $\mathbf{A}$ consists of the eigenvectors of

$$\mathbf{W}^{-1}\mathbf{B}. \tag{4.45}$$

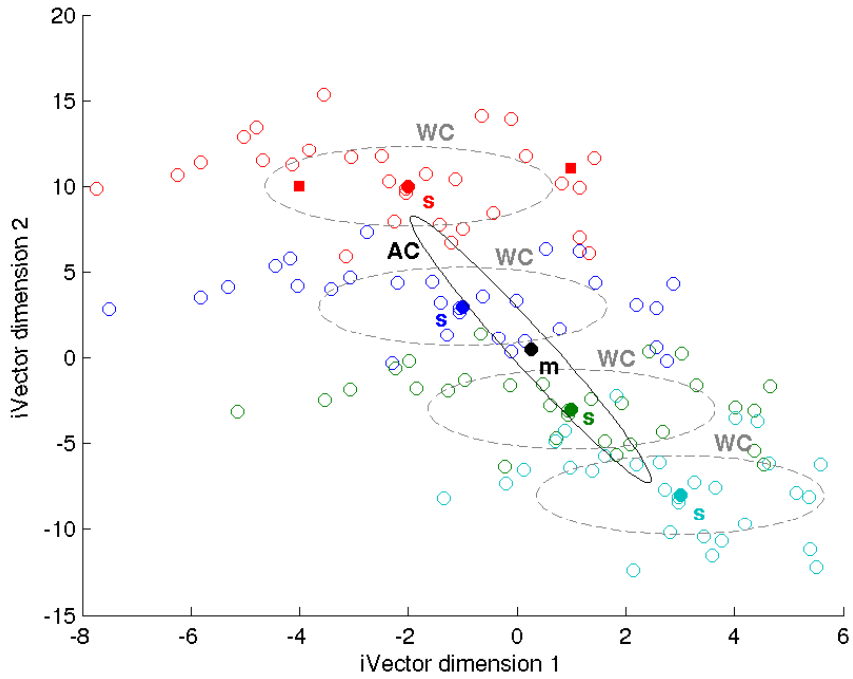


Figure 4.1: LDA assumption in iVector space.

Usually $\mathbf{A}$ is sorted according to the corresponding eigenvalues. This way, meaningful dimensionality reduction can be performed by keeping simply the n leading basis.

With a LDA matrix $\mathbf{A}$ computed using iVectors from a background set and the speaker labels, each iVector $\mathbf{w}$ can be transformed first by the LDA. Next, the WCCN matrix $\hat{\mathbf{W}}^{-1}$ is trained to whiten the within-class covariance matrix of the LDA transformed feature space. It is estimated using 4.44 on the transformed iVectors. Incorporating both into the cosine scoring, equation 4.42 can be rewritten as:

$$score_{\texttt{LDA+WCCN}} = \frac{\mathbf{w}_1^T \mathbf{A} \hat{\mathbf{W}}^{-1} \mathbf{A}^T \mathbf{w}_2}{\sqrt{\mathbf{w}_1^T \mathbf{A} \hat{\mathbf{W}}^{-1} \mathbf{A}^T \mathbf{w}_1} \sqrt{\mathbf{w}_2^T \mathbf{A} \hat{\mathbf{W}}^{-1} \mathbf{A}^T \mathbf{w}_2}}. \tag{4.46}$$

As this is not a likelihood-ratio, normalization is needed, usually zt-norm.

However, this approach has been significantly outperformed by the use of a Probabilistic Linear Discriminant Model (PLDA) [Prince, 2007] to model speaker and session variability in iVector space. As this model will mainly be used for the iVector based experiments, it is explained in more detail in the following section.

4.3.3 Probabilistic Linear Discriminant Analysis

The fixed-length iVectors extracted per utterance can be used as input to a pattern recognition algorithm. Here, a probabilistic framework is used, where speaker and intersession variability in the iVector space is modeled using across-class and within-class covariance matrices $\mathbf{\Sigma}_{ac}$ and $\mathbf{\Sigma}_{wc}$. It is assumed that latent vectors $\mathbf{s}$ representing speakers are distributed according to

$$p(\mathbf{s}) = \mathcal{N}(\mathbf{s}|\mathbf{m}, \mathbf{\Sigma_{ac}}) \tag{4.47}$$

and for a given speaker $\mathbf{s}$, the iVectors are distributed as

$$p(\mathbf{w}|\mathbf{s}) = \mathcal{N}(\mathbf{w}|\mathbf{s}, \mathbf{\Sigma_{wc}}). \tag{4.48}$$

Figure 4.1 exemplifies this assumption by a toy example in 2D iVector space. The dots represent several iVectors extracted from several utterances corresponding to four different speakers. It can be observed, that the four solid dots, representing the explicit speakers means $\mathbf{s}$, are Gaussian distributed with mean $\mathbf{m}$ and covariance $\mathbf{\Sigma}_{ac}$. Further, the individual iVectors per speaker are also Gaussian distributed with its mean $\mathbf{s}$ and a globally tied covariance $\mathbf{\Sigma}_{wc}$. Using this model, it becomes clear that two new data points (red squares) are likely to belong to the same speaker although they are quite far apart from each other. As it is a probabilistic model, we can compute likelihoods of data and the parameters $\mathbf{m}$, $\mathbf{\Sigma}_{ac}$ and $\mathbf{\Sigma}_{wc}$ can be estimated using ML (EM algorithm).

Further, in the Probabilistic Linear Discriminant Analysis Model (PLDA) [Prince, 2007], it is assumed that the speaker and/or intersession variability can be modeled using subspaces. PLDA can be seen as a special case of JFA ([Kenny et al., 2008b], Section 4.3.1) with a single Gaussian component. Here, a simplified variant (SPLDA) is used, which we adapted from [Brümmer, 2010]. The across-class covariance matrix is modeled using a subspace as $\mathbf{\Sigma}_{ac} = \mathbf{V}^T\mathbf{V}$, which limits the speaker variability to live in a subspace spanned by the columns of the reduced rank matrix $\mathbf{V}$. With full-rank within-class covariance matrix $\mathbf{\Sigma}_{wc} = \mathbf{D}^{-1}$, the distribution of the iVectors $\mathbf{w}$ can be rewritten as:

$$\mathbf{w} = \mathbf{m} + \mathbf{V}\mathbf{y} + \boldsymbol{\varepsilon} \tag{4.49}$$

incorporating a low-dimensional speaker subspace $\mathbf{V}$ and $\boldsymbol{\varepsilon}$ being a noise variable (covering the channel effects) with precision matrix $\mathbf{D}$.

Similar to the JFA model, the latent variables $\mathbf{y}$ are distributed according to a zero-mean unit-covariance Gaussian prior:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{I}), \quad p(\boldsymbol{\varepsilon}) = \mathcal{N}(\boldsymbol{\varepsilon}|\mathbf{0}, \mathbf{D}^{-1}). \tag{4.50}$$

ML and minimum divergence update of parameters

Now, a simplified version of an EM algorithm [Brümmer, 2010] with minimum divergence update is presented, to estimate the SPLDA model parameters.

The global mean parameter $\mathbf{m}$ is simply defined by the mean of all iVectors $\mathbf{w}_{su}$ in the training data, with $s = 1 \dots S$ speakers and $u = 1 \dots U_s$ observations per speaker s. The zero order statistics per speaker are simply U_s and the first order statistics are defined as:

$$\mathbf{f}_s = \sum_{u=1}^{U_s} \mathbf{w}_{su} - \mathbf{m}. \tag{4.51}$$

The global second order statistic is:

$$\mathbf{S} = \sum_{u=1}^{SU} (\mathbf{w}_{su} - \mathbf{m})(\mathbf{w}_{su} - \mathbf{m})^T. \tag{4.52}$$

The posterior of the speaker factors given all the observations $\mathbf{W}_s = [\mathbf{w}_{s1} \dots \mathbf{w}_{su_s}]$ is normally distributed with:

$$p(\mathbf{y}_s|\mathbf{W}_s) = \mathcal{N}(\mathbf{y}_s|\hat{\mathbf{y}}_s, \mathbf{P}^{-1}), \tag{4.53}$$

where the precision matrix of the posterior is

$$\mathbf{P}_s = U_s(\mathbf{V}^T\mathbf{D}\mathbf{V}) + \mathbf{I} \tag{4.54}$$

and the mean of the posterior distribution is given as

$$\hat{\mathbf{y}}_s = \mathbf{P}_s^{-1}\mathbf{V}^T\mathbf{D}\mathbf{f}_s. \tag{4.55}$$

Two helper variables $\mathbf{Q}$ and $\mathbf{R}$ are introduced that are accumulated over all speakers:

$$\mathbf{Q} = \sum_s \hat{\mathbf{y}}_s \mathbf{f}_s^T, \tag{4.56}$$

$$\mathbf{R} = \sum_s U_s(\mathbf{P}^{-1} + \mathbf{y}_s\mathbf{y}_s^T). \tag{4.57}$$

Further, for the minimum divergence update

$$\mathcal{Y} = \frac{1}{S}\sum_{s=1}^{S}(\mathbf{P}^{-1} + \mathbf{y}_s\mathbf{y}_s^T) \tag{4.58}$$

is accumulated.

Using the complete-data likelihood function for speaker s

$$p(\mathbf{W}_s|\mathbf{y}_s, \mathbf{V}, \mathbf{D}) = \prod_{u=1}^{U_s} \mathcal{N}(\mathbf{w}_{su}|\mathbf{V}\mathbf{y}_s, \mathbf{D}^{-1})$$
$$\propto \exp \sum_{u=1}^{U_s} (-\frac{1}{2}\mathbf{w}_{su}^T\mathbf{D}\mathbf{w}_{su} + \mathbf{w}_{su}^T\mathbf{D}\mathbf{V}\mathbf{y}_s - \frac{1}{2}\mathbf{y}_s^T\mathbf{V}^T\mathbf{D}\mathbf{V}\mathbf{y}_s), \tag{4.59}$$

we can obtain the EM auxiliary function to maximize

$$\langle \sum_s \log p(\mathbf{W}_s|\mathbf{y}_s, \mathbf{V}, \mathbf{D}) + \text{const} \rangle$$
$$= \frac{N}{2} \log |\mathbf{D}| - \frac{1}{2}\text{tr}(\mathbf{SD}) - \frac{1}{2}\text{tr}(\mathbf{RV}^T\mathbf{DV}) + \text{tr}(\mathbf{QDV}). \quad (4.60)$$

The maximum likelihood update for $\mathbf{V}$ is then given by:

$$\mathbf{V} = \mathbf{R}^{-1}\mathbf{Q} \quad (4.61)$$

and $\mathbf{D}^{-1}$ can be updated as

$$\mathbf{D}^{-1} = \frac{1}{N}(\mathbf{S} - \mathbf{VQ}) \quad (4.62)$$

with N being the global zero order statistics $\sum_{s=1}^{S} U_s$. Finally, minimum divergence re-estimation of $\mathbf{V}$ gives:

$$\mathbf{V} \leftarrow \mathbf{V}\texttt{chol}(\mathcal{Y})^T \quad (4.63)$$

where $\texttt{chol}(\mathcal{Y})\texttt{chol}(\mathcal{Y})^T = \mathcal{Y}$ denotes the Cholesky decomposition. This scheme is used in an iterative manner that usually converges in less than ten iterations.

Score evaluation for a pair of iVectors

After parameter estimation, the SPLDA model can be directly used to evaluate speaker trials incorporating two iVectors $\mathbf{w}_1$ and $\mathbf{w}_2$ due to its probabilistic nature. Following the work of [Burget et al., 2011] the needed formulae are provided below. For a more general view, allowing multiple enrollment iVectors, see [Brümmer and de Villiers, 2010].

Using the trained SPLDA model and Equations 4.47 and 4.48, one can directly evaluate the log-likelihood ratio between two hypotheses:

1. H_s: Both iVectors were generated from (a mode of) a single speaker.
2. H_d: The two iVectors were generated independently (from two different speakers).

The LLR is computed as:

$$LLR_{\texttt{PLDA}} = \log \frac{p(\mathbf{w}_1, \mathbf{w}_2|H_s)}{p(\mathbf{w}_1, \mathbf{w}_2|H_d)} = \log \frac{\int p(\mathbf{w}_1|\mathbf{s})p(\mathbf{w}_2|\mathbf{s})p(\mathbf{s})d\mathbf{s}}{p(\mathbf{w}_1)p(\mathbf{w}_2)}. \quad (4.64)$$

The numerator gives the marginal likelihood of producing both iVectors from the same speaker, while the denominator is the product of the marginal likelihoods that both iVectors are produced independently. The integrals can be evaluated analytically and scoring can be performed very efficiently:

$$LLR_{\texttt{PLDA}} = \mathbf{w}_1^T\mathbf{\Lambda}\mathbf{w}_2 + \mathbf{w}_2^T\mathbf{\Lambda}\mathbf{w}_1 + \mathbf{w}_1^T\mathbf{\Gamma}\mathbf{w}_1 + \mathbf{w}_2^T\mathbf{\Gamma}\mathbf{w}_2 + (\mathbf{w}_1 + \mathbf{w}_2)^T\mathbf{c} + k \quad (4.65)$$

with

$$\mathbf{\Gamma} = -\frac{1}{4}(\mathbf{\Sigma}_{wc} + 2\mathbf{\Sigma}_{ac})^{-1} - \frac{1}{4}\mathbf{\Sigma}_{wc}^{-1} + \frac{1}{2}\mathbf{\Sigma}_{tot}^{-1}, \tag{4.66}$$

$$\mathbf{\Lambda} = -\frac{1}{4}(\mathbf{\Sigma}_{wc} + 2\mathbf{\Sigma}_{ac})^{-1} + \frac{1}{4}\mathbf{\Sigma}_{wc}^{-1}, \tag{4.67}$$

$$\mathbf{c} = ((\mathbf{\Sigma}_{wc} + 2\mathbf{\Sigma}_{ac})^{-1} - \mathbf{\Sigma}_{tot}^{-1})\mathbf{m}, \tag{4.68}$$

$$k = \log|\mathbf{\Sigma}_{tot}| - \frac{1}{2}\log|\mathbf{\Sigma}_{wc} + 2\mathbf{\Sigma}_{ac}| - \frac{1}{2}\log|\mathbf{\Sigma}_{wc}| + \mathbf{m}^T(\mathbf{\Sigma}_{tot}^{-1} - (\mathbf{\Sigma}_{wc} + 2\mathbf{\Sigma}_{ac})^{-1})\mathbf{m}, \tag{4.69}$$

and the total covariance matrix given by

$$\mathbf{\Sigma}_{tot} = \mathbf{\Sigma}_{ac} + \mathbf{\Sigma}_{wc} = \mathbf{V}^T\mathbf{V} + \mathbf{D}^{-1} \tag{4.70}$$

with $\mathbf{\Sigma}_{wc} = \mathbf{D}^{-1}$ (see 4.62) and $\mathbf{\Sigma}_{ac} = \mathbf{V}^T\mathbf{V}$ (see 4.63).

4.3.4 Experiments

Both techniques, the JFA as well as the iVector modeling, are applicable to the low-dimensional well-defined DCT features as presented in Section 3.

The initial baseline system was a standard UBM-GMM system with MAP adaptation [Reynolds et al., 2000]. In Section 3.1.5, results for this basic system, using the simple DCT contour features were presented. An EER of around 25% on the 2006 NIST SRE task was obtained.

Joint Factor Analysis for DCT contour features

Prior to the NIST Speaker Recognition Evaluations 2008, the JFA approach as described in Section 4.3.1 could be applied to the same prosodic contour features leading to significant improvements [Kockmann and Burget, 2008b].

First, we will describe how the JFA model is trained. The starting point is again a UBM. We train two gender dependent models with 256 Gaussian components each and diagonal covariances, using the background data set (see Section 2.2.1). Further, for each utterance in the background set, zero and first order sufficient statistics are collected using the final UBM.

Then, the JFA model itself can be trained as described in Algorithm 1. The mean vector $\mathbf{m}$ is set to the concatenated UBM means. The used mean supervector dimension is $256 \times 13 = 3328$ and the speaker-subspace size is chosen as $R_{\mathbf{V}} = 50$ and the channel-subspace size $R_{\mathbf{U}} = 20$, similar to the setup in [Dehak et al., 2007]. Both matrices $\mathbf{U}$ and $\mathbf{V}$ are randomly initialized and are retrained in 10 iterations. Note, that only speakers, that have at least five sessions in the background data set are used.

Following Algorithm 1, with all JFA hyperparameters fixed, the speaker models can be trained and evaluated. Again, the 2006 NIST SRE core condition is used, and for each training utterance, speaker- and channel factors $\mathbf{y}$ and $\mathbf{x}$ are estimated jointly by stacking $\mathbf{U}$ and $\mathbf{V}$ into a single 70-dimensional subspace.

Norm	Male	Female	Both
none	18.23	16.27	17.12
z-norm	18.48	15.64	16.47
t-norm	15.83	13.90	14.73
zt-norm	15.69	13.35	14.37

(a) Standard LLR

Norm	Male	Female	Both
none	15.96	14.17	15.00
z-norm	16.22	15.45	15.28
t-norm	14.08	11.53	12.40
zt-norm	14.63	13.07	12.95

(b) Fast scoring

Table 4.1: DCT contour features with pseudo-syllable segmentation modeled by JFA with two different scoring techniques. Results in EER [%] for SRE 2006 core condition.

Finally, channel factors $\mathbf{x}$ are estimated for each test utterance in the NIST SRE 2006 data set. As mentioned earlier, we estimate the channel factors for the test utterances based on the UBM only by setting $\mathbf{y}$ to zero.

The first line in Table 4.1.a shows great improvements achieved due to the JFA modeling with conventional LLR scoring. While around 25% EER is obtained with the standard GMM-UBM approach, the EER can be significantly reduced to 17%.

Next, also score normalization techniques as described in [Auckenthaler et al., 2000] are applied. For this purpose, several hundred z- and t-norm utterances are randomly taken from the background data set. While z-norm already improved the performance, a great improvement is obtained by using t- or even zt-norm (meaning first to apply z- and then t-norm). While z-norm parameters can be pre-computed, especially t-norm is computationally expensive, as several hundred extra trials per test utterance have to be evaluated to obtain the normalization statistics.

The fast-scoring technique that reduces the likelihood evaluation to a dot-product scoring has already been introduced in Section 4.3.1. Especially for the evaluation of a big t-norm cohort (each t-norm speaker has to be scored against each test utterance), this technique seems to be very appropriate. As the results in Table 4.1.b indicate, the technique is not only much faster, but seems to be even more robust and results in significantly better performance. It is also interesting to see, that both prosodic JFA systems work reasonably well without any normalization, which is usually not the case using low-level cepstral features.

Beside these results, many more experiments were conducted during the work on this thesis, mainly investigating into:

- Optimal subspace sizes: Different from cepstral systems, the optimal number of eigenvoices $R_{\mathbf{V}}$was found to be 50–100 and the number of eigenchannels $R_{\mathbf{U}}$ 20–40.

- Optimal model sizes: Depending on the amount of training data, the optimal number of Gaussian components C was 256–1024.

- Amount of training data: While increasing the amount of data for the UBM does not improve the performance much, a rule of thumb for the JFA model can be: "The more, the better".

- JFA training style: Joint estimation of **U** and **V** [Kenny et al., 2005] or iterative re-estimation (also taking channel into account when estimating the speaker matrix) did not improve the performance.

- Minimum-divergence training: As proposed by [Kenny et al., 2008b], did not improve the performance but achieves a faster convergence and requires less training iterations.

- Scoring with integration over the channel distribution: A technique used in the original JFA system proposed by [Kenny et al., 2005] performs equally to the fast-scoring technique, but is much slower.

iVector modeling for DCT contour features

During NIST SRE 2010 system development [Brümmer et al., 2010], we tried to apply the iVector approach as described in Section 4.3.2 to the DCT contour features, as it greatly outperformed the standard JFA approach on cepstral systems [Dehak et al., 2009a].

Both models, JFA and iVector models, were trained and compared. For both, the staring points are gender dependent UBMs trained on the background data. As the NIST SRE 2008 data set was used for development (see Section 2.2.3), the NIST 2006 data set was included into the background data set. Due to the larger amount of data, larger models with 1024 Gaussian components were used. Again, sufficient statistics are collected for each utterance in the background set, which are used for training the JFA as well as for training the iVector extractor.

The JFA model was trained exactly as described in the previous section with 100 eigenvoices and 40 eigenchannels per gender. The total variability model is trained in a similar manner. The total variability matrix $\mathbf{T}$ with a size $R_{\mathbf{T}} = 100$ is randomly initialized and is trained in 10 iterations of the algorithm as described in Section 4.3.2.

Then, the iVector model parameters are fixed and the model can be used as a front-end to collect iVectors for all training and test utterances. This is first done for each utterance in the background set. Based on these iVectors, a deterministic LDA+WCCN channel compensation scheme as described in Section 4.3.2 was computed. Using the speaker labels, within class and across class covariance matrices are computed based on the iVectors to obtain a LDA transformation, reducing the dimensionality of the iVectors to 75. Based on the transformed 75-dimensional iVectors, a full-rank WCCN matrix is computed and applied.

When comparing the results on EER in Table 4.2, very disappointing results for the iVector approach with syllable based segmentation and cosine distance scoring are obtained. This iVector configuration achieves results that are about 40% relative worse than the standard JFA model.

Model	Segmentation	Scoring	EER
JFA	Syllable	Fast scoring	11.97
iVector	Syllable	LDA 75 + WCCN + Cosine distance	16.86
JFA	Fixed	Fast scoring	9.36
iVector	Fixed	LDA 75 + WCCN + Cosine distance	12.12
iVector	Fixed	PLDA 100	8.42

Table 4.2: JFA vs. iVector modeling for different DCT contour feature segmentations and scoring methods on SRE 2008 telephone condition. All with zt-norm, except the PLDA model.

However, in recent experiments [Kockmann et al., 2011c] we have shown that the iVector modeling itself is superior to the JFA approach. Mainly two things changed compared to the above experiments. First, the fixed window size segmentation as proposed in Section 3.1.2 was used with a window size of 300ms and a frame-shift of only 50ms. It was found, that while already good results were obtained compared to the pseudo-syllable segmentation (see results in Section 3.1.5), reducing the frame-shift from 150ms to 50ms greatly improved the performance.

For a comparable JFA system as used in the last experiments, this already improved the general system performance, as can be seen in the third row of Table 4.2, leading to an EER of under 10% on the NIST 2008 *tel-phn:tel-phn* condition. Note, that many more frames are extracted using this segmentation technique due to the high overlap of consecutive windows. This seems to be generally helpful for the system performance. However, we did not systematically optimize the frame-shift size.

Using these features for an iVector model with LDA+WCCN as in the latter experiments, still resulted in worse results for iVector approach, but decreased the EER from 16% to 12%. New was to apply the PLDA model for trial evaluation, as described in Section 4.3.3. We used a full-rank 100-dimensional speaker subspace $\mathbf{V}$ and no further normalization. This greatly improved the performance resulting in an EER of 8.4%, 10% relative better than the JFA system. Note, that score normalization techniques were not found to be helpful for the PLDA model. The best scores were always obtained without any score normalization, indicating that the scores are already well distributed due to the probabilistic nature and evaluation of the LLR scores.

4.4 Subspace models for parameters of multinomial distributions

As described in the last section, there were successful attempts to apply JFA and recently also iVector modeling to prosodic features. However, only a small subset of well-defined continuous prosodic features, as the proposed DCT contour features, can be used with JFA.

In [Ferrer et al., 2010], the two most popular prosodic speaker verification approaches are compared:

1. JFA modeling of a subset of prosodic contour features.

2. Discretizing (binning) a full set of SNERF prosodic features, representing it as a vector of soft counts and modeling it using SVM.

Further, the two modeling techniques are compared using exactly the same subset of contour features, once modeling the features directly using JFA and once converting the features to soft counts and modeling them using SVMs. Although JFA compares favorably to SVM on the subset of well-defined continuous contour features, a significant gain can be obtained with SVMs trained on count super-vectors based on SNERFs (see Section 3.2) when prosodic features – that JFA cannot deal with – are added.

In this section, an approach is proposed to combine the advantage of the JFA-like subspace model with the flexibility of representing highly complex heterogeneous prosodic SNERF features.

A similar idea of subspace modeling of multinomial distribution was proposed for inter-session variability compensation in phonotactic language identification [Glembek et al., 2008]. A related model is also applied for modeling GMM weights in Subspace GMM (SGMM), which is a recently proposed acoustic model for speech recognition [Povey and Burget, 2011].

4.4.1 Total variability subspace

Based on the success of the iVector approach for cepstral features, there was a motivation to develop the simplified JFA-like variant with only a single total variability subspace for SNERFs. As described in Section 3.2, the SNERFs are parameterized to discrete class counts based on GMMs.

As a generative model, a multinomial distribution appears as a natural choice for modeling the counts resulting from this step. More precisely, a set of E multinomial distributions is required, one for each GMM in the ensemble. Each multinomial distribution corresponds to a set of C_e probabilities, one probability ϕ_{ec} for each Gaussian c in the GMM e. For each frame, each GMM is expected to generate a feature by one of its components with probability given by the multinomial distribution. This corresponds to co-occurring events that should be modeled by separate multinomial distributions (as all SNERF feature-tokens are modeled independently of each other). Each multinomial distribution lives in a n-dimensional simplex and the space of all parameters is the Cartesian product of all the simplexes for all the separate distributions. The bottom row of Figure 4.2 illustrates this for the toy example already used when SNERFs were introduced in Section 3.2. Remember, three prosodic measurements are extracted (first row) and parameterized by individual small GMMs (second row). Now, as each mixture component represents a discrete class, a multinomial distribution is obtained for each prosodic feature. The parameters of the duration model exist on a line; the pitch model parameters, in a 2D simplex; and the energy parameters, in a 3D simplex space.

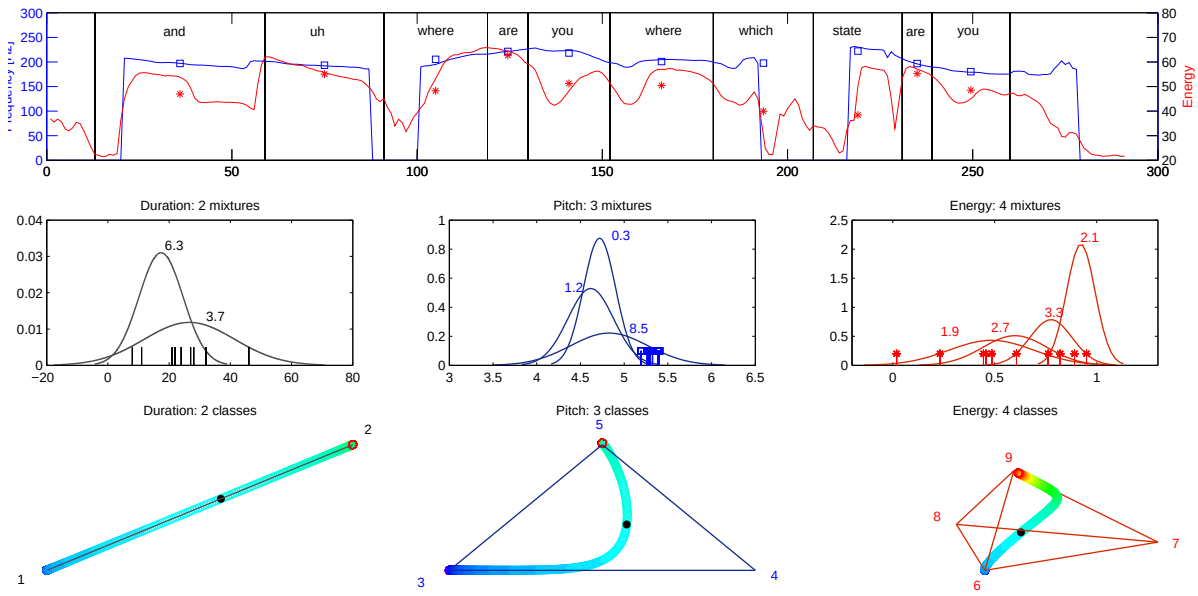


Figure 4.2: **Top row:** Extraction of three SNERF parameters from a speech segment containing 10 single-syllable words: Syllable duration (determined by black vertical lines), mean pitch value per syllable (blue squares), and mean energy per syllable (red stars). **Middle row:** Parameterization of SNERF sequences: Small GMMs are trained on background data for each individual SNERF. Two mixtures are used for duration, three mixtures for pitch, and four mixtures for energy. Occupation counts for the values extracted in the top row (here as bars) are collected using the GMMs. **Bottom row:** Multinomial model spaces for duration, pitch, and energy. The colored lines show various one-dimensional iVectors (the values are mapped to colors) projected to the full ensemble of multinomial spaces.

In the proposed Subspace Multinomial Model (SMM), it is assumed that the multinomial distributions differ from utterance to utterance. In the case of SNERFs, parameters of many multinomial distributions need to be estimated. Therefore, a way to estimate all the parameters robustly given a limited amount of data available for each utterance is desired. It is assumed that there is a low-dimensional subspace of the parameter space in which the parameters for individual utterances live.

For this reason, an explicit latent variable $\mathbf{w}$ is introduced through which the probability of cth class of multinomial distribution e in the ensemble is:

$$\phi_{ec} = \frac{\exp(m_{ec} + \mathbf{t}_{ec}\mathbf{w})}{\sum_{i=1}^{C_e} \exp(m_{ei} + \mathbf{t}_{ei}\mathbf{w})}, \tag{4.71}$$

with $\mathbf{t}_{ec}$ being the cth row of eth block of a low-dimensional subspace matrix $\mathbf{T}$ (size $\sum_{e=1}^{E} C_e \times R_{\mathbf{T}}$), which spans a linear subspace that is generally non-linear in the simplex of class probabilities due to the softmax function. Figure 4.2 shows how the subspace restricts the movement in the simplex in a non-linear way (colored lines). By drawing values for

a one-dimensional variable $\mathbf{w}$ from minus infinity to infinity, the model moves in all three simplexes simultaneously along the non-linear, single-dimensional manifolds.

Now, all the multinomial distributions corresponding to one utterance can be represented by a single low-dimensional vector $\mathbf{w}$. This way, (1) the number of free parameters to efficiently model differences between individual utterances can be reduced, and (2) dependencies between the individual SNERFs can be learned.

The model parameters are estimated by iteratively re-estimating the latent variables $\mathbf{w}$ for each utterance in the training data to maximize the likelihood function based on the current estimate of $\mathbf{T}$, and vice-versa.

Using the final estimate of $\mathbf{T}$, $\mathbf{w}$ vectors (which are also called iVectors) can be extracted for new data. This way, the model is used as a feature extractor and each iVector can be seen as a low-dimensional representation of the whole utterance.

Next, re-estimation formulae for the proposed subspace model are presented. For simplicity, a single multinomial distribution is used and we show how to extend the framework to the general case with arbitrary number of multinomial distributions. While only the final update formulae needed for implementation are given in this section, the whole derivation can be found in Appendix B.

Likelihood function

The log-likelihood of data $\mathbf{O}$ for a multinomial model with C discrete classes is determined by the vector of model parameters $\boldsymbol{\phi}$ and vector of sufficient statistics $\boldsymbol{\gamma}$, representing the occupation counts of classes for all U utterances in $\mathbf{O}$:

$$\log p(\mathbf{O}) = \sum_{u=1}^{U} \sum_{c=1}^{C} \gamma_{uc} \log \phi_{uc}, \tag{4.72}$$

where γ_{uc} is the occupation count for class c and utterance u and ϕ_{uc} are the probabilities of the (utterance dependent) multinomial distribution, which is defined by a subspace model according to Equation 4.71:

$$\phi_{uc} = \frac{\exp(m_c + \mathbf{t}_c \mathbf{w}_u)}{\sum_{i=1}^{C} \exp(m_i + \mathbf{t}_i \mathbf{w}_u)}, \tag{4.73}$$

where m_c is a speaker independent mean parameter, $\mathbf{t}_c$ is the c-th row of subspace matrix $\mathbf{T}$ and $\mathbf{w}_u$ is an $R_{\mathbf{T}}$-dimensional column vector (iVector) representing speaker and channel of utterance u.

Parameter re-estimation

The model parameters are obtained by maximum likelihood (ML) estimation. First, the subspace parameters $\mathbf{m}$ and $\mathbf{T}$ need to be estimated from training data. This is an iterative process, alternating between estimating subspace parameters $\mathbf{m}$ and $\mathbf{T}$ with fixed iVectors, and estimating iVectors $\mathbf{w}_u$ (one for each training utterance) with fixed subspace

parameters. Even with fixed subspace parameters, there is no closed-form solution for the ML update of iVectors, and each iVector must be updated using an nonlinear optimization technique, which is again an iterative procedure. Likewise, there is no closed-form solution for the ML update of the subspace parameters with fixed iVectors.

An efficient iterative optimization scheme based on Newton methods [Fletcher, 2000] can be applied which uses a local quadratic approximation to the log likelihood function. In Appendix B, the full derivations for a Newton-Raphson update scheme based on the log-likelihood function 4.72 are given.

However, the final updates adopted in our implementation are based on updates derived for Subspace GMMs [Povey and Burget, 2011]. These differ mainly in the way that an approximate Hessian is used, and lead to better convergence. Another alternative would be to use quasi Newton methods [Jorge and Stephen, 2006, Chapter 6] like the BFGS algorithm, that do not need to evaluate the Hessian matrix explicitly.

In the used implementation, vectors $\mathbf{w}_u$ are updated as

$$\mathbf{w}_u^{new} = \mathbf{w}_u^{old} + \mathbf{H}_u^{-1}\mathbf{g}_u, \tag{4.74}$$

where $\mathbf{g}_u$ is the gradient of the log likelihood function

$$\mathbf{g}_u = \sum_{i=1}^{C} \mathbf{t}_i^T(\gamma_{ui} - \phi_{ui}^{old}\sum_{j=1}^{C}\gamma_{uj}) \tag{4.75}$$

and $\mathbf{H}_u$ is an $R_\mathbf{T} \times R_\mathbf{T}$ matrix

$$\mathbf{H}_u = \sum_{i=1}^{C} \mathbf{t}_i^T\mathbf{t}_i \max(\gamma_{ui},\ \phi_{ui}^{old}\sum_{j=1}^{C}\gamma_{uj}), \tag{4.76}$$

where ϕ_{ui}^{old} refers to the multinomial distribution (4.73) defined by the parameters from the preceding iteration. Note that the matrix $\mathbf{H}_u$ can be interpreted as an approximation to the Hessian matrix and the update formula (4.74) can be then seen as a Newton-Raphson update. The rows of matrix $\mathbf{T}$ are updated as

$$\mathbf{t}_c^{new} = \mathbf{t}_c^{old} + \mathbf{H}_c^{-1}\mathbf{g}_c, \tag{4.77}$$

where $\mathbf{g}_c$ is the gradient of the log likelihood function

$$\mathbf{g}_c = \sum_{u=1}^{U}(\gamma_{uc} - \phi_{uc}^{old}\sum_{i=1}^{C}\gamma_{ui})\mathbf{w}_u^T \tag{4.78}$$

and $\mathbf{H}_c$ is an $R_\mathbf{T} \times R_\mathbf{T}$ matrix

$$\mathbf{H}_c = \sum_{u=1}^{U} \max(\gamma_{uc},\ \phi_{uc}^{old}\sum_{i=1}^{C}\gamma_{ui})\mathbf{w}_u\mathbf{w}_u^T. \tag{4.79}$$

The updates for both $\mathbf{w}_u$ and $\mathbf{T}$ may fail to improve the likelihood by making too large an update step. In the case of such failure, the update step is halved until an increase in likelihood is obtained.

Vector $\mathbf{m}$ is usually set to the logarithm of the normalized counts of the whole background data set and kept fix during re-estimation and is not updated. However, retraining $\mathbf{m}$ can be simulated by fixing one of the coefficients in vectors $\mathbf{w}_u$ to be one and setting the corresponding column of matrix $\mathbf{T}$ as the vector $\mathbf{m}$. However, We never saw any benefit from further retraining $\mathbf{m}$.

Algorithm 2 Quasi-algorithm for SMM training and prosodic iVector extraction.

```
{Estimate total variability subspace}
m ← log of mean of background data probabilities
T ← PCA of 4.80 (or random)
w ← 0
for i = 1 → number of outer iterations do
  for j = 1 → number of inner w iterations do
    for all utterances in background data do
      Update w per utterance (4.74)
    end for
  end for
  for k = 1 → number of inner T iterations do
    for all rows in T do
      Update t (4.77)
    end for
  end for
end for
{Estimate iVectors}
w ← 0
for j = 1 → number of inner w iterations do
  for all utterances in background/enrollment/test data do
    Update w per utterance (4.74)
  end for
end for
```

So far, only subspace modeling of single multinomial distribution is considered in the equations. However, for the prosodic features extracted by the ensemble of GMMs, the occupation counts should be modeled by a set of multinomial models, one for each GMM. These are considered to be concatenated into a single super-vector of multinomial distributions, which is modeled by one subspace matrix $\mathbf{T}$. In other words, there will be only one iVector $\mathbf{w}_u$ defining the whole set of multinomial distributions for each utterance u. To achieve this, the indices c from Equation (4.73) must be divided into subsets, where each subset corresponds to mutually exclusive events (counts from one GMM). Then, the only difference will be in the denominator of (4.73), where the normalization will be per-

formed only over the appropriate subset of indices that the current c belongs to (just like in Equation 4.71).

After the subspace parameters are estimated on training data, the model can be used to extract iVectors $\mathbf{w}_u$ for all enrollment, test and background utterances using the same update formulae (4.74–4.76). The whole training process is exemplified in Algorithm 2.

Model initialization

While Section 4.4.1 contains a quite general description of the training procedures, the model initialization is described here more specifically for the used system. First, the multinomial distributions for individual GMMs from the ensemble are estimated using all training utterances. This corresponds to summing all training super-vectors of occupation counts and normalizing the resulting super-vector over the ranges corresponding to the individual GMMs. Such super-vector of multinomial distributions (holding the individual class probabilities) is denoted as $\mathbf{sv}_{UBM}$. The vector $\mathbf{m}$ is simply initialized to a log of $\mathbf{sv}_{UBM}$. Note, that no advantage from its further retraining was observed using the updates from the previous section. All vectors $\mathbf{w}$ are initialized to zeros. To ensure a good starting point, the subspace matrix $\mathbf{T}$ is initialized to the eigenvectors of the covariance matrix computed from smoothed utterance super-vectors $\mathbf{sv}_u$ centered around the vector $\mathbf{m}$, where coefficient c of vector $\mathbf{sv}_u$ is defined as:

$$sv_{uc} = \log(\alpha \frac{\gamma_{uc}}{f_{uc}} + (1-\alpha) sv_{UBM_{uc}}), \tag{4.80}$$

where f_{uc} is the number of feature frames seen for the utterance and for the GMM that the occupation count γ_{uc} corresponds to. The smoothing constant $\alpha = 0.9$ ensures that the log of zero is not taken for classes that have not been occupied at all by any frames of utterance u.

4.4.2 Experiments

Training algorithm

First experiments are performed to evaluate the convergence properties of three possible ways to train the subspace model:

1. Gradient Descent (GD): Subspace and iVectors are solely updated using the gradient of the error function with a fixed learning rate parameter η: In Equations 4.74 and 4.77, the Hessian term is substituted with a fixed parameter η.

2. Newton-Raphson, Iterative Reweighted Least Squares (IRLS): A full IRLS update is used for subspace and iVectors as described in Appendix B. Note, that the evaluation of the full Hessian matrix is needed, which is infeasible for large subspaces.

3. Hessian approximation (HA): This approximation of the Hessian is adopted from [Povey and Burget, 2011] (Equations 4.74–4.79). It is claimed that this leads to better convergence properties than the full IRLS.

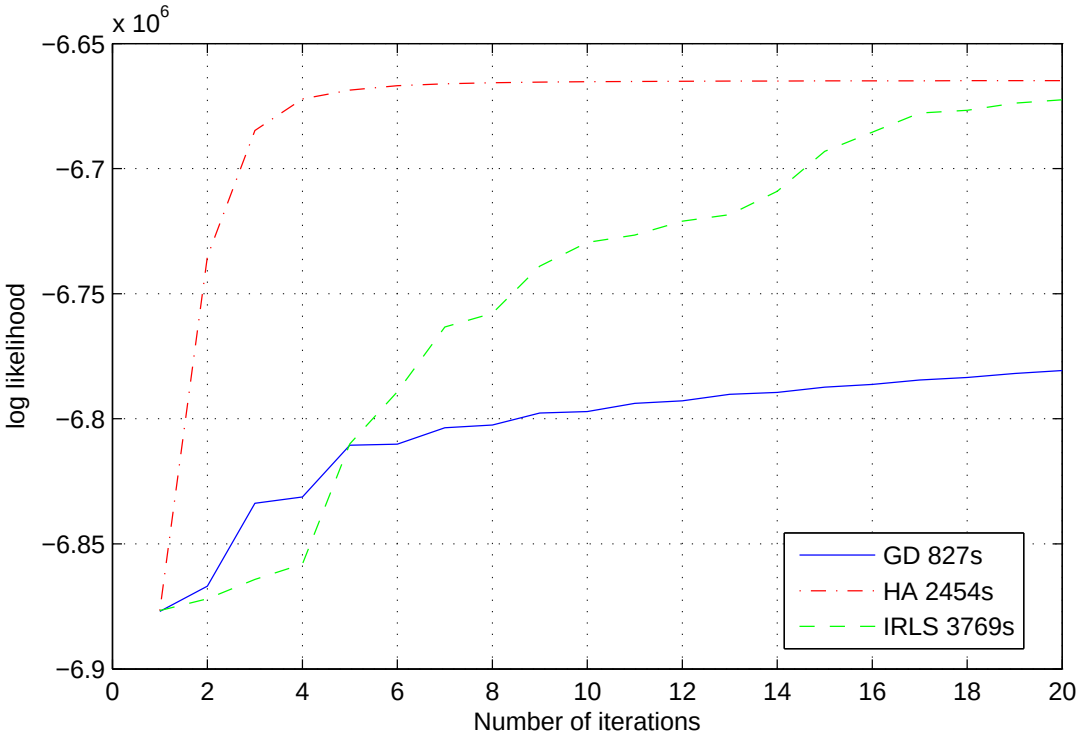


Figure 4.3: Training of a SMM based on SNERFS with $45,000 \times 20$-dimensional subspace on $1,500$ utterances with three different approaches: Gradient Descent (GD), Newton-Raphson (IRLS) and the one adopted from Subspace GMMs (HA).

Figure 4.3 shows the log-likelihood of the training data for several iterations of different training algorithms. As the full IRLS is computationally very demanding, a small subspace of size $R_{\mathbf{T}} = 20$ is trained on only 1,500 training utterances. The SNERF count vectors for each utterance are approximately 45,000-dimensional. While the GD algorithm is the fastest to perform 20 training iterations, it can be seen that it converges quite slowly. Note, that the fixed stepsize η has to be halved several times during the first iterations.

Interestingly, the same phenomena is observed for the full Newton-Raphson training (IRLS) as mentioned in [Povey and Burget, 2011]. While it converges properly in the end, the stepsize has to be halved during the training several times. Due to this and the general high computational demand, the IRLS is the slowest algorithm.

The adopted approximation of the Hessian generally results in the best convergence properties. The algorithm seems to converge in five iteration. Due to the approximation of the Hessian, it can be still computed for large subspace matrices with many classes.

SNERF SVM baseline

First results for the proposed model were published in [Kockmann et al., 2010a] based on a cooperative work with SRI. SRI provided the SNERF features to train and test a

speaker verification system for the NIST SRE 2006 task. In the experiments, the underlying baseline system is a remake of the SRI SNERF system as described in detail in [Ferrer et al., 2007].

First, the provided SNERFs were parameterized as described in Section 3.2. After training of the background models, Gaussian component occupation counts were gathered for each training and test utterance. All counts were divided by the number of frames and further rank-normalized to serve as high-dimensional input features for an SVM classifier. LIBSVM [Chang and Lin, 2001] is used to train a linear kernel SVM using all background utterances as negative examples. Each speaker model SVM is then used to classify the test utterance counts, and the output is used as a decision score.

Further, t-norm is applied to the scores. Without the score normalization, an EER of 13.3% could be achieved and t-norm decreases the EER to 12.7%, as shown in Table 4.3. The results are in agreement with those reported for the reference system [Ferrer et al., 2007].

Subspace size

Next, the new modeling approach itself is evaluated on the same data as the SVM baseline. First, alternate training of $\mathbf{w}$ and $\mathbf{T}$ according to Section 4.4.1 is used in three iterations to increase the likelihood on the training data. Once $\mathbf{T}$ is trained, vectors $\mathbf{w}$ for all background, training, and test utterances are estimated in one iteration. Vectors $\mathbf{w}$ are used as input for an SVM with cosine kernel as proposed in [Dehak et al., 2009b]. No better accuracy was seen than for the linear kernel, but the rank normalization (which was used in the baseline system with the linear kernel) could be omitted. The cosine kernel is used also for all following experiments. Also, no significant change in EER was observed with t-norm applied to the scores of the subspace model, so all results reported for this model are without t-norm.

Figure 4.4 shows the trend of EER for different numbers of factors $R_{\mathbf{T}}$. While the performance is bad with small number of factors, the EER decreases and converges quickly and indicates that the proposed approach is indeed working. Interestingly, with 250 factors, better accuracy than the baseline system for males (12.4% EER) is achieved, but is worse for females (14.7% EER) (EER of the baseline system in Table 4.3 is slightly better for the female than for males).

Another interesting property is, that reasonably good performance could be obtained when the subspace $\mathbf{T}$ is initialized purely with Principal Component Analysis (PCA). In preliminary experiments, an absolute loss of only 1% in EER was achieved with this highly simplified approach.

Intersession compensation with LDA+WCCN

On top of the low-dimensional input vectors, intersession compensation is performed using the methods described in [Dehak et al., 2009b] and Section 4.3.2, based on the best-performing subspace system with 250 factors.

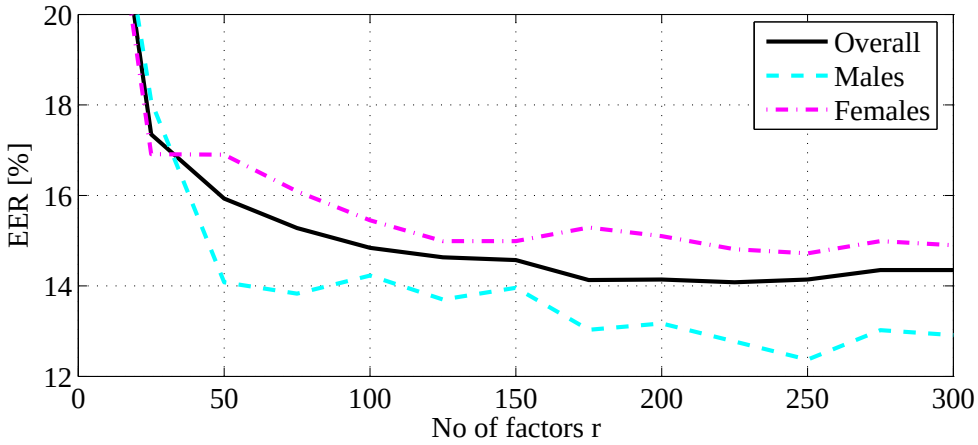


Figure 4.4: Dependence of EER on size of subspace for a SMM based on SNERFs.

Classifier	Overall		Males		Females	
	EER	DCF	EER	DCF	EER	DCF
Full SVM baseline, linear kernel + t-norm	12.72	5.07	12.90	4.57	12.61	5.38
SVM cosine kernel	14.14	5.34	12.37	4.64	14.72	5.73
SVM WCCN kernel	11.32	4.61	9.98	3.91	11.86	5.01
SVM LDA(75) + WCCN kernel	9.91	4.90	9.44	4.28	10.33	5.26
LDA(75) + WCCN + Cosine distance + zt	8.83	4.86	9.04	4.28	8.69	5.25
PLDA(100)	**7.21**	**3.59**	**6.78**	**3.12**	**7.06**	**3.86**

Table 4.3: EER and DCF (×100) for SNERF baseline and SMM subspace systems with 250 factors and different classifiers on the SRE 2006 core condition.

First, an WCCN matrix is trained on the labeled background data iVectors. The WCCN matrix is then directly integrated into the SVM cosine kernel (similar to Equation 4.46, without the LDA matrix $\mathbf{A}$), which is used on the low-dimensional iVectors extracted for the NIST 2006 evaluation data. As depicted in Table 4.3, a relative improvement of 20% in EER and 14% in DCF is achieved, respectively, due to the WCCN.

Alternatively, LDA+WCCN is used to diagonalize the across-class-covariance matrix and make the within-class-covariance matrix the identity matrix. LDA can be used to further reduce the feature dimension by dropping the nuisance directions that correspond to the channel. As shown in Table 4.3, reducing the iVector dimensionality by LDA seems to help a lot and the best error rate of 9.9% is achieved, using LDA(75) reduction to 75 dimensions.

Finally, a simplification of the speaker enrollment and testing procedure is used, also successfully applied in [Dehak et al., 2009b] (and described in Section 4.3.2), where the

value of the SVM kernel function evaluated for enrollment and test utterance is directly taken as the score (Equation 4.46). This greatly reduces the computational complexity, as no SVM training is needed and only a cosine distance is computed during testing. As shown in the fifth line of Table 4.3, a further improvement of 11% relative to an excellent EER of 8.8% is obtained with this fast scoring technique. Also, the observed difference between the male and female subsystems seems to vanish. Note, that zt-norm had to be applied to these scores to obtain good performance, while no positive effect of zt-norm was seen for the SVM-based approaches (However, we need a cohort of negative examples to train the speaker model SVMs).

Another interesting phenomenon that was observed in the experiments is, that while the EER was consistently better for a smaller number of dimensions (50-100 after LDA), the DCF was generally better for a larger number of dimensions.

Intersession compensation with PLDA

Further experiments were carried out to investigate the use of the PLDA model (as described in Section 4.3.3) for trial evaluation and intersession compensation.

We compared the PLDA framework on exactly the same SRE 2006 setup as used in the latter section. The last row of Table 4.3 shows the results for a PLDA model with 100-dimensional speaker subspace $\mathbf{V}$, without further normalization. The PLDA model significantly outperforms all other approaches in terms of EER and DCF.

Further experiments were published in [Kockmann et al., 2011b]. The experiments are performed on the NIST 2008 *tel-phn:tel-phn* data set and the PLDA model is first trained on the background set, including the NIST 2006 data. The Subspace Multinomial Model iVector system as well as the SNERF counts are equivalent to the previously described experiments. The full rank PLDA model is trained on 200-dimensional iVectors using the algorithm described in Section 4.3.3. We used PLDA scoring (Section 4.3.3) without any score normalization. In these experiments, the PLDA modeling reached an EER of 6.9% and a normalized minimum DCF of 0.33., again significantly outperforming the cosine distance scoring with LDA as used in the previous experiments with SMMs (around 9% EER) [Kockmann et al., 2010a].

5 Final comparative study

In this chapter, a final experimental evaluation of the prosodic feature extraction and modeling techniques developed during the thesis will be presented. Diverse prosodic speaker verification systems and a novel combination of those [Kockmann et al., 2011c] are presented. Further, we show how the developed systems can be combined with a cepstral baseline system leading to significant improvements.

Results are presented on the NIST 2008 development set (see Section 2.2.3) and on the NIST 2010 evaluation set (see Section 2.2.4) for all three defined conditions:

1. *tel-phn:tel-phn*
2. *int-mic:tel-phn*
3. *int-mic:int-mic*

These involve telephone as well as interview speech recorded over various microphones in a room. All UBMs, JFA, iVector or PLDA models are trained on exactly the same lists incorporating a mixture of conversational telephone and interview microphone speech from Switchboard and NIST SRE 2004–2008 corpora (as described in Chapter 2 and further defined in [Scheffer et al., 2010]).

Beside DET plots incorporating minimum DCF values, also actual DCF measures are presented for selected systems which allows us to assess the quality of the calibration for the evaluation set (see Section 2.1 for a description).

5.1 Results for prosodic systems

5.1.1 System descriptions

DCT-JFA

The first system is similar to the one used in the ABC submission to NIST SRE 2010 evaluation [Brümmer et al., 2010] and is based on findings presented in Sections 3.1.5 and 4.3.4. This system uses a simple prosodic feature set as described in Section 3.1 and the JFA modeling approach as described in Section 4.3.1.

The extracted pitch and energy values are modeled by DCT over a fixed length window of 300ms with a window shift of 50ms. The first 6 DCT coefficients for the pitch and energy trajectories are used per segment. Prior to DCT approximation, all unvoiced frames (where no pitch is detected) are cut out. The number of voiced frames is further appended as a duration feature. This way, well-defined 13 dimensional feature vectors are obtained every 50 ms.

A separate multivariate 13 dimensional UBM is trained per gender, with 512 components and diagonal co-variances. Note, that we apply variance flooring, meaning that variances are set to a certain value if they become too small.

As a next step, a separate JFA model is trained per gender. The JFA mean vector $\mathbf{m}$ is set to the mean supervector of the UBM and $\mathbf{V}$ and $\mathbf{U}$ are initialized randomly. First, subspace $\mathbf{V}$ is estimated in ten iterations. Next, a single 50-dimensional channel subspace $\mathbf{U}$ is obtained in another ten iterations of EM, with fixed estimate of $\mathbf{V}$. The residual term $\mathbf{Dz}$ is not used in the setup as it never showed any improvements. These subspace sizes were found optimal during the NIST SRE 2010 system development [Brümmer et al., 2010].

With all background models fixed, the speaker models are enrolled for the development and evaluation sets. This is done by estimating the 100 dimensional speaker factor vectors $\mathbf{y}$ per enrollment utterance. Next, the channel factors $\mathbf{x}$ have to be estimated per test utterance.

Having all parameters of the JFA model for each trial, the log-likelihood ratio for each trial can be evaluated. Again, the computationally efficient method based on the sufficient statistics [Glembek et al., 2009] is used. Finally, zt-norm [Auckenthaler et al., 2000] is applied, using impostor enrollment and test utterances taken from the background set.

DCT-iV

The second system is similar to the `DCT-JFA` in terms of features as well as subspace modeling. However, the simplified variant of JFA is used, using a single total variability as described in Section 4.3.2. The same algorithmic framework is used to estimate the subspace as for the JFA model, but now the subspace model serves as a feature extractor for another probabilistic backend. The system is based on exactly the same DCT features, UBMs and sufficient statistics as the first system. Next, a separate 300 dimensional total variability subspace $\mathbf{T}$ is trained for each gender. The same EM algorithm as for the JFA is used, again with 10 iterations.

Using the final estimate of $\mathbf{T}$, 300-dimensional total variability vectors (iVectors) are extracted, for all background, enrollment and test utterances.

The extracted iVectors can now be used to evaluate scores for trials based on a PLDA model, as described in Section 4.3.3: Vector $\mathbf{m}$ is set to global mean of the training iVectors. Across-class covariance matrix $\mathbf{\Sigma}_{ac}$ and within-class covariance matrix $\mathbf{\Sigma}_{wc}$ are modeled with full rank of 300 and the parameters are iteratively re-estimated using an EM algorithm as described in Section 4.3.3. No further score-normalization is performed.

SNERF-iV

The third single prosodic system is based on features which are completely different from the previous ones. Heterogeneous high-dimensional SNERF features as described in Section 3.2 are used. First, 182 basic SNERFs are extracted per syllable of each utterance. 9 different n-gram tokens up to order three are formed for the SNERFs, incorporating also pauses in speech.

Next, gender-dependent UBMs are trained for these features on all background data, but instead of training a single high-dimensional multivariate GMM per gender, a separate low-dimensional GMM with a low number of mixture components is trained for each SNERF and n-gram. This way, 1,638 separate GMMs are trained per gender. The number of mixture components per GMM depends on the relative frequency of the n-gram tokens and varies between 3 and 186. As some SNERF features might be undefined for a syllable (i. e. when no pitch is detected or the syllable lacks onset), each UBM is trained with a standard EM algorithm using only defined features first. In the second step, the models are retrained using a special variant of EM [Kajarekar et al., 2004] that can handle undefined values and estimates special parameters for these (see also Section 3.2.4). The final estimate of the UBMs are used to extract zero order sufficient statistics for all utterances.

These soft counts are used to train a SMM with a single total variability subspace as described in Section 4.4.1. The full vector of statistics incorporate counts from 1,638 individual UBMs and has a dimensionality of 45,818. A 300-dimensional subspace is used to capture the most relevant information from the statistics. Again, a subspace size of 200–300 has been found to be optimal in earlier experiments, as can also be observed in Figure 4.4. The SMM parameter vector $\mathbf{m}$ is set to log of the concatenated UBM weights and the subspace matrix $\mathbf{T}$ is initialized randomly. The latent variables $\mathbf{w}$ are set to zero vectors for each utterance. Using this initialization, three iterations of the optimization scheme described in Section 4.4.1 are done. Within each outer iteration a single inner iteration is run, first for $\mathbf{w}$ and then for $\mathbf{T}$.

The final estimate of $\mathbf{T}$ is used to extract 300-dimensional total variability vectors (iVectors), for all background, enrollment and test utterances in another single iteration of the optimization scheme.

These iVectors are then used to train a PLDA backend with full-rank covariance matrices $\mathbf{\Sigma}_{ac}$ and $\mathbf{\Sigma}_{wc}$ in the same way as for the previous system. The score for each trial is obtained using PLDA trained on SNERF iVectors. No score normalization is used.

DCT+SNERF-iV

As both prosodic iVector systems perform very well, and at the same time are significantly different in terms of features as well as the modeling approach, a combination of both seems natural. Since both modeling techniques translate the long-temporal prosodic feature vectors of variable size to a single fixed-length iVector per utterance, it is possible to simply concatenate the iVectors resulting from these diverse models and to model them jointly with a single PLDA model. The 300 dimensional iVector pairs from the `DCT-iV` system

and the SNERF-iV system are simply concatenated for each utterance and 600- dimensional DCT+SNERF iVectors are obtained for all the data.

Then, a single full-rank PLDA model is trained on the 600-dimensional iVectors using the same technique as for the single iVector systems. The PLDA model is then used to evaluate the speaker trials on the concatenated 600 dimensional iVectors.

5.1.2 *tel-phn:tel-phn* condition

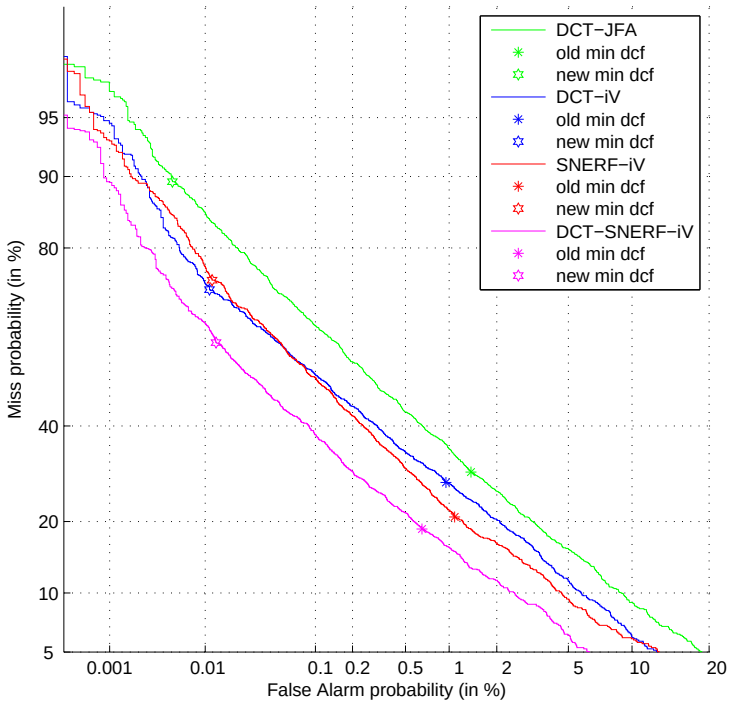
Figure 5.1 shows DET plots for the *tel-phn:tel-phn* conditions for the NIST SRE 2008 development set and the NIST SRE 2010 evaluation set. In these conditions, all utterances are conversational telephone speech recordings. The plots include curves for all three single prosodic systems; the old and new DCF points are marked as defined for NIST SRE 2008 (old) and NIST SRE 2010 (new). The green line in Figure 5.1.a shows the performance of the DCT-JFA system on the development set. An EER of 9.36% and an old normalized DCF of 0.433 is achieved. The new normalized DCF point is generally very high for the prosodic systems (close to the maximum of 1).

Using the iVector approach, with a PLDA backend instead of JFA modeling for the DCT contour features, consistent and significant improvements are achieved. The blue line shows the performance for the DCT-iV system. Relative improvements of up to 20% is achieved due to the iVector frontend and PLDA backend.

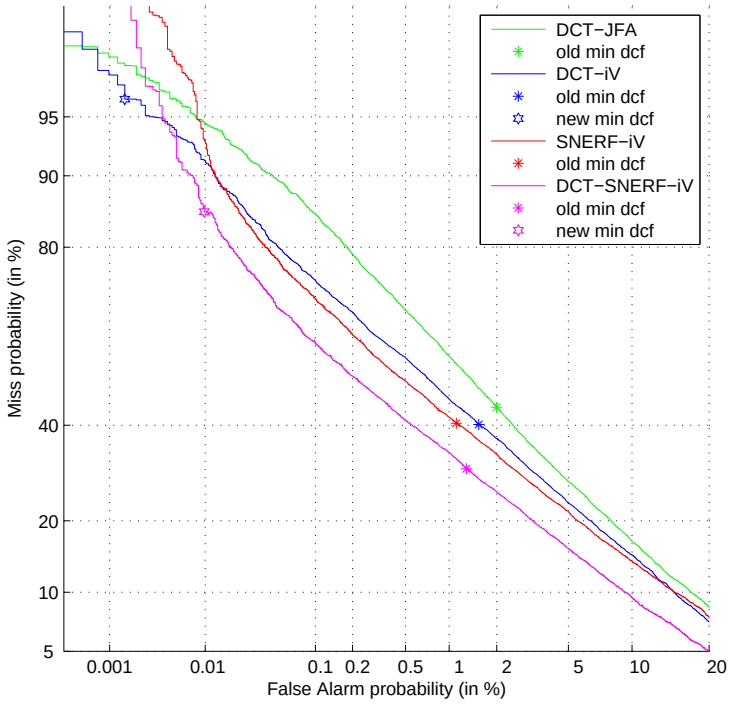
The third system, the SNERF-iV system, is shown in red. This system gives the best performance in terms of EER and old DCF, with 7.0% EER and an old DCF of 0.316. However, the two iVector systems are quite close in performance and the DET curves even cross several times in the low false alarm regions.

A similar trend can be observed for the evaluation set in Figure 5.1.b. The DCT-JFA system gives the worst performance, then the DCT-iV and the SNERF-iV systems. Again, the two iVector systems are quite close to each other and the SNERF-iV system performs worse in the low false alarm regions. The red curve for the SNERF-iV system shows a strange behavior with rapid performance degradation in the low false alarm region. Generally, the system performance is worse on the 2010 data than on the 2008 data. The best achieved EER lies around 12% and the best old DCF around 0.5. The new DCF points are often close to one and are not even depicted in the plot.

The effectivity of the joint modeling of complementary iVectors can be observed in Figure 5.1 on magenta curves. For 2008 and 2010 data, significant gains over the best single system can be obtained on most operating points. With an EER of 5.4%, an old DCF of 0.252 and a new DCF of 0.730, relative improvements of up to 20% are obtained on 2008 data. On 2010 data an EER of 9.7%, an old DCF of 0.429 and a new DCF of 0.953 are achieved. These are also relative improvements of up to 20% over the best single system.



(a) DEV 2008



(b) EVAL 2010

Figure 5.1: DET plots for *tel-phn:tel-phn* condition for three different prosodic systems plus combination of the two iVector systems on development and evaluation sets.

5.1.3 *int-mic:tel-phn* condition

Next, the behavior of the system performance for trials not involving only conversational telephone speech, but also speech recorded over different microphones in an interview scenario is analyzed. Note, that the systems are exactly the same as before. There are no JFA, iVector or PLDA models trained for a certain condition.

In the *int-mic:tel-phn* condition all enrollment utterances come from an interview scenario where the interviewee is recorded over several close-talk and distant microphones. The test utterances are again conversational speech recorded over telephone channels.

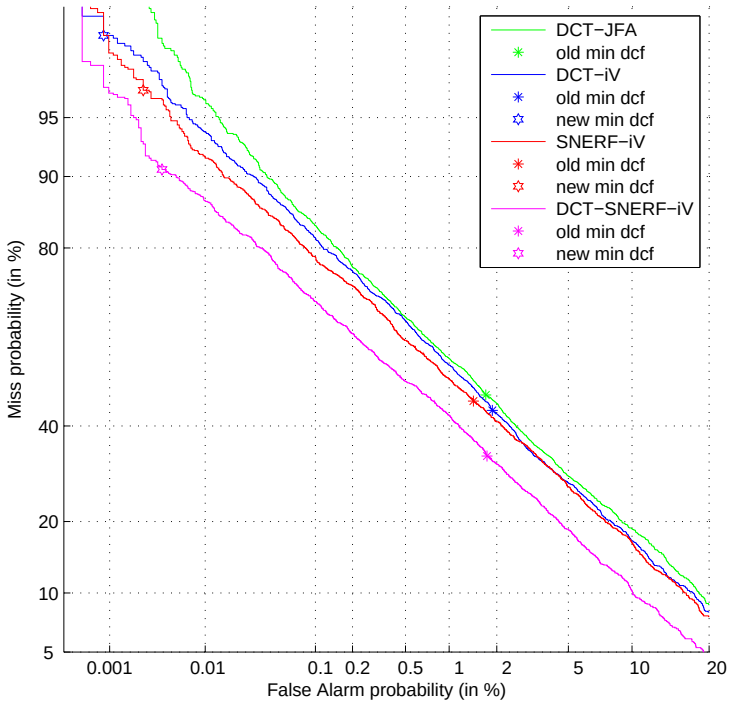
Figure 5.2 shows the DET plots for the *int-mic:tel-phn* condition for the two test sets. In subplot (a), a similar trend can be observed as for the previous condition. The `DCT-JFA` systems performs the worst with an EER of 13.87% and an old DCF of 0.64. As expected, the general performance on this condition is lower than for the matched condition with telephone conversations in training and test. As a consequence, the new DCF measure is again close to the maximum of one (not plotted). The two iVector systems both outperform the JFA system, with the `SNERF-iV` system being the better one, especially in the very low false alarm area. However, the difference between the three prosodic systems is much smaller than for the `tel-phn:tel-phn` condition. Still, high gains are achieved from the iVector combination of `DCT-iV` and `SNERF-iV` features. EER is reduced to 10% (21% relative) and old DCF is reduced to 0.5 (17% relative).

The trend of 2010 data being more challenging also continues for this condition as can be seen in Figure 5.2.b. Also, all three system stay very close in terms of performance. Surprisingly the `SNERF-iV` system results in the worst EER, followed by `DCT-JFA` and `DCT-iV` with the best EER of 15.0%. Moving to the low false alarm area, the `SNERF-iV` system moves closer to the `DCT-iV` system, outperforming the `DCT-JFA` system. All old DCF points lie very close to each other in the range from 0.67–0.71 and the new DCF points are again too close to the maximum to be plotted. Still, for this condition, the `DCT-iV` system remains the best single prosodic system. The combined `DCT-SNERF-iV` system also outperforms the individual systems on this test set. An EER of 13.5% and an old DCF of 0.592 reflect relative gains of up to 12%. Here, the gains are generally lower than what is observed on the 2010 *tel-phn:tel-phn* condition.

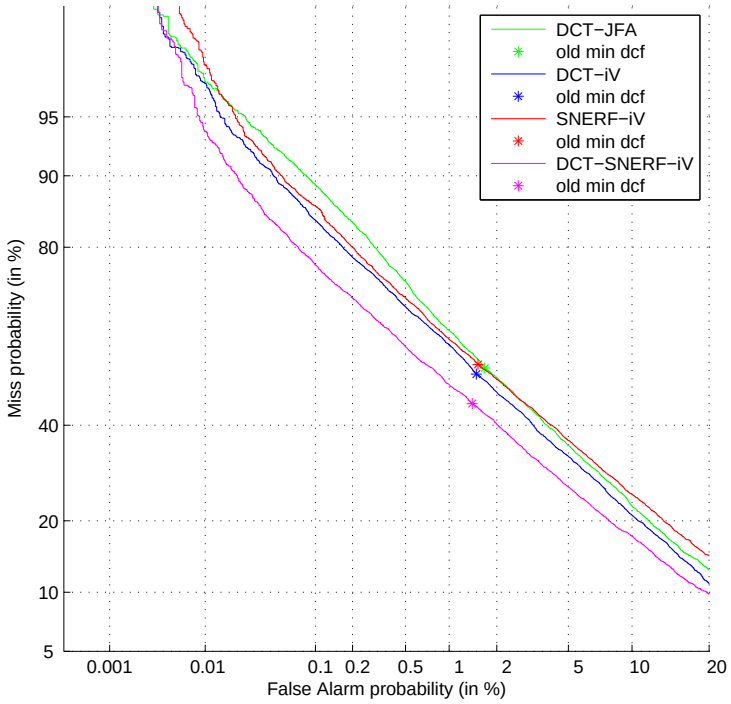
5.1.4 *int-mic:int-mic* condition

The last conditions to be presented are the *int-mic:int-mic* conditions for 2008 and 2010 data sets. Now, both training and test recordings come from an interview scenario. Still, different microphone types are used. Again, exactly the same models as for the previous two conditions are used.

Figure 5.3.a presents the results on 2008 data for the three individual systems, as well as the iVector combination, as a DET plot. Consistent to the previous findings, the `DCT-JFA` system gives the worst general performance. Still, at the EER point it performs the same as the `SNERF-iV` with 11.8%. However, at the old DCF point, the `SNERF-iV` system outperforms the `DCT-JFA` system, but getting worse again in the very low false

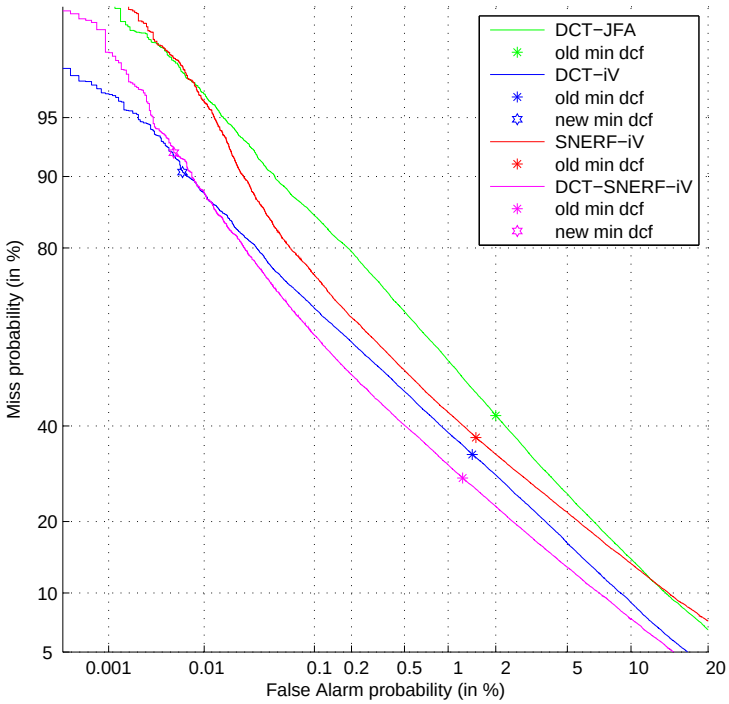


(a) DEV 2008

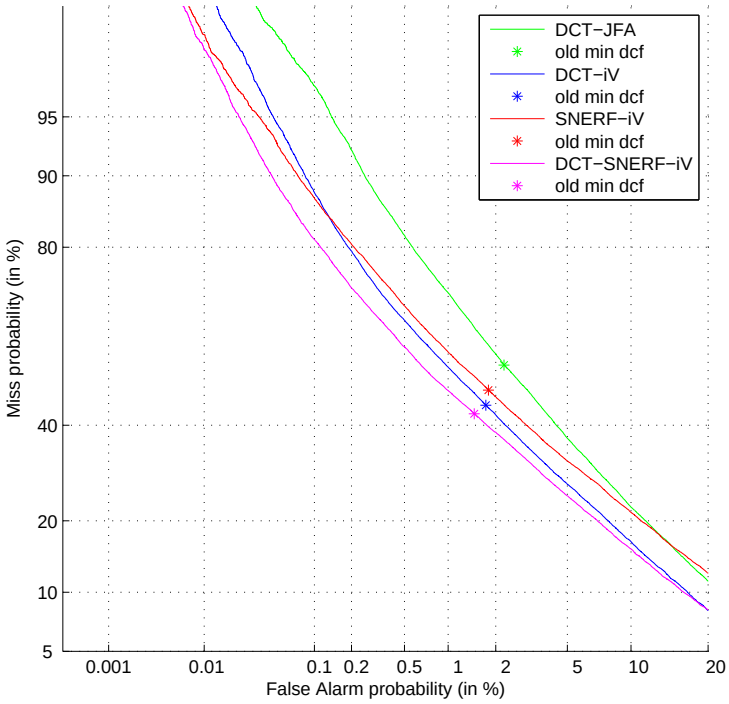


(b) EVAL 2010

Figure 5.2: DET plots for *int-mic:tel-phn* condition for three different prosodic systems plus combination of the two iVector systems on development and evaluation sets.



(a) DEV 2008



(b) EVAL 2010

Figure 5.3: DET plots for *int-mic:int-mic* condition for three different prosodic systems plus combination of the two iVector systems on development and evaluation sets.

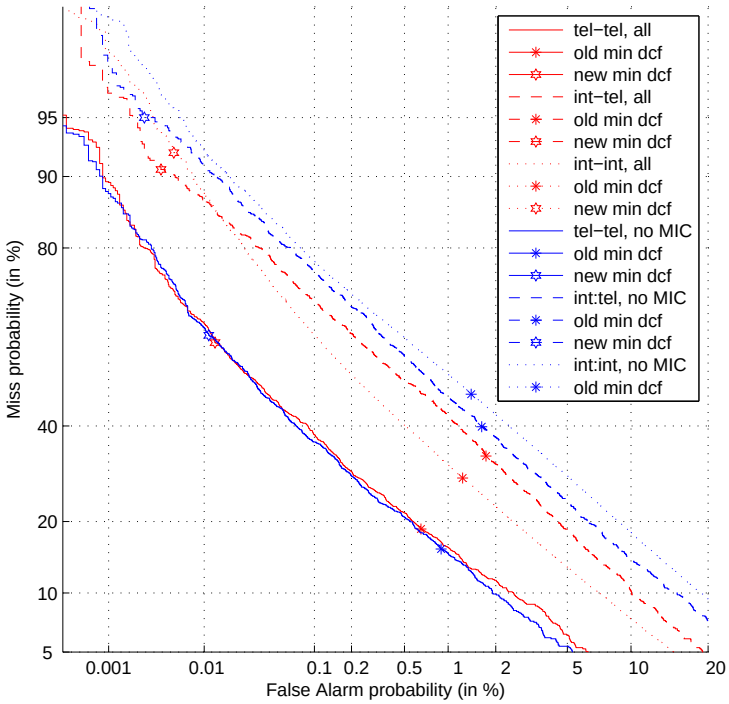
alarm region. Once more, the DCT-iV system outperforms the others; on this condition consistently and significantly on all operating points, with up to 20% relative in terms of EER. Generally, better results than for the second (unmatched) condition are obtained. The iVector combination again outperforms the other systems with an EER of 8.6% and an old DCF of 0.401. In the low false alarm region, a behavior that also occurred for the other conditions can be observed. The SNERF+DCT iVector fusion curve seems to run parallel to the single SNERF-iV system. As a consequence, also the fusion leads to degraded performance around the new DCF point, being worse than the single DCT-iV system.

Results on 2010 data in Figure 5.3.b again confirm that the iVector modeling followed by PLDA consistently outperforms the standard JFA approach. With an EER of 13% and an old DCF of 0.62, the iVector approach outperforms the JFA modeling by 15–19% relative. The SNERF-iV system again shows an inconsistent behavior. While it gives the worst EER, it outperforms the DCT-iV system towards the new DCF point. Still, the iVector combination gives consistent gains over the single systems. As for the 2008 data, the gains are not that high on this condition.

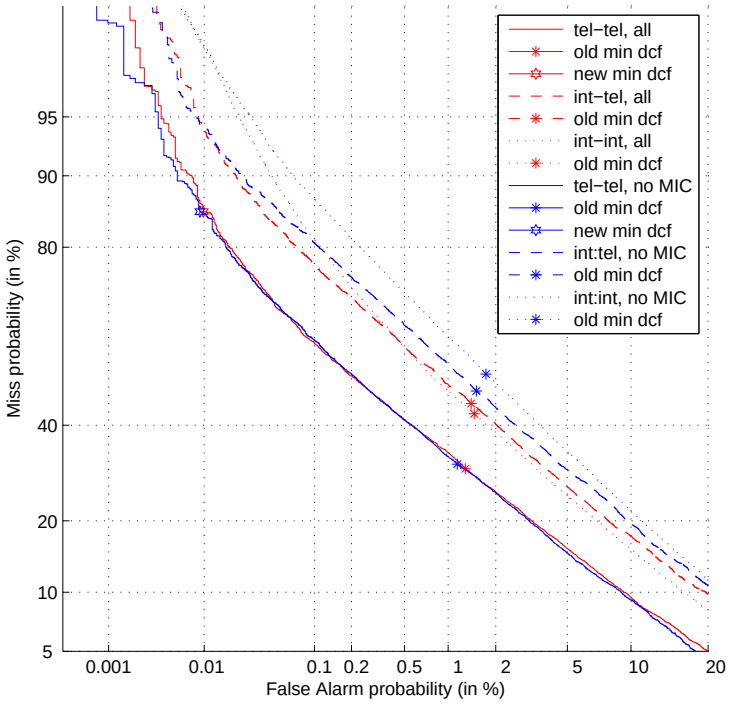
5.1.5 Final observations

Several observations could be made during the final experiments. First, a single system trained on all kinds of speech style and recording types seems to work reasonable well under different conditions. No separate subspace models for the different conditions are trained, no tuned feature extraction or any special normalization tricks are used. All the processing steps are completely equal for all three tested conditions. Figure 5.4 shows the effect of the type of PLDA training data for the different conditions for the best performing prosodic SNERF+DCT-iV system. The DET plot shows all three conditions (solid, dashed, and dotted lines), once the PLDA is trained on all data (red lines) and once trained only using telephone data (blue lines). One can see, that there is no significant change on *tel-phn:tel-phn* condition when incorporating only telephone data into the training, so adding data from other conditions does not seem to affect the performance. On the other hand, excluding the microphone speech from the PLDA training degrades the performance on the two conditions incorporating microphone speech. Especially for the *int-mic:int-mic* condition, relative degradations of up to 60% were observed (from the red dotted to the blue dotted line).

This might also explain that best results are obtained on *tel-phn:tel-phn* followed by *int-mic:int-mic* and then *int-mic:tel-phn*. The system is mainly optimized for telephone speech and most training material for the subspace models comes from telephone conversations. Still, a lot of interview speech recorded over several microphones for the same speakers is used in the training set to learn these channel conditions. For the *int-mic:tel-phn* condition, there is a lack of proper training data. There are only few recordings from the same person of conversational telephone speech and of interview speech recorded over microphones. So, the model can learn the channel space for matched conditions well (like for different types of telephones or microphones), but not the channel space for unmatched conditions (between telephones and microphones).



(a) DEV 2008



(b) EVAL 2010

Figure 5.4: DET plots showing the effect of PLDA training with telephone data only (no MIC) and all data including microphone data (all) on all three conditions (tel-tel,int-tel and int-int) for `SNERF+DCT-iV` system on development and evaluation sets.

As a second finding, the new DCF point does not seem to be very meaningful for the performance of the prosodic systems. Often, it is very close to the maximum of one and the DET plot gets very steppy in this range. As a result, inconsistent behavior in the very low false alarm region of the DET plots is often observed.

Generally, it can be observed that the iVector frontend followed by PLDA is consistently superior to the standard JFA model for the simple DCT features. This holds for both test sets and all conditions. So far, the `SNERF-iV` system was the best performing system during the experiments for this thesis. While the best overall performance on the *tel-phn:tel-phn* condition is still achieved with the `SNERF-iV` system, this does not hold for the other two conditions incorporating microphone speech. Except for 2008 *int:mic-tel-phn* condition, the `DCT-iV` system outperforms the `SNERF-iV` system. It can be observed that for the SNERF-iV system, the DET curve tends to indicate worse performance on many conditions around the new DCF point. This is something that we can not explain now and that needs further investigations. Anyway, we should point out that the DCT features are much simpler than the SNERFs and much easier and faster to extract. Modeling of GMM mean parameters seem to be more robust and powerful than the modeling of the multinomial distributions.

However, the combination on the iVector level for the two iVector systems remains superior on all relevant operating points. Consistent gains of up to 20% are achieved with respect to the best single system on all conditions. The *iVector fusion* seems to be an appropriate method to combine complementary iVector frontends.

As a last finding, it can be said that the 2010 data seems to be generally more challenging than the 2008 data set.

5.2 Calibration

Up to now, we have not investigated how the systems perform in a real-world scenario, where an accept/reject threshold has to be set before new data is processed. This fact is taken into account in the *actual DCF* measures (see Section 2.1.3). While the minimal DCF measures the costs for an optimal chosen threshold, the actual DCF measures the costs for unseen data with a previously chosen threshold.

Calibration is a post-processing of the system output scores, so that these scores are interpretable as proper log-likelihood ratios. As we will use a linear calibration, we assume that shifting and scaling of the scores is appropriate for this. After training of the calibration parameters using a development set and the desired target prior, we can make decisions to minimize the risk using Bayes decision theory by comparing the calibrated scores to a Bayes decision threshold [Brümmer and de Villiers, 2011]

$$\log \frac{C_{\text{FalseAcceptance}}}{C_{\text{Miss}}} - \log \frac{P_{\text{Target}}}{1 - P_{\text{Target}}} \tag{5.1}$$

using the costs and target prior as defined in Section 2.1.

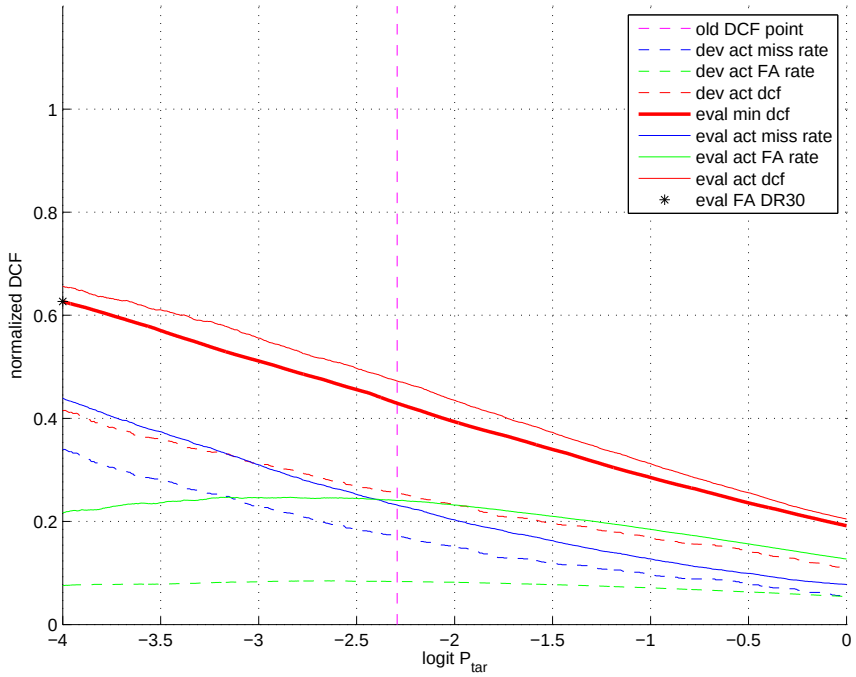
To investigate this, the best performing `SNERF+DCT-iV` system is calibrated on each of the three conditions of the 2008 development set separately, and the learned parameters are applied to the matched condition of the 2010 evaluation set. Note, that scores from a PLDA should be already interpretable as log-likelihood ratios, so it should be possible to set a threshold analytically to make decisions for optimizing DCF. However, supervised training of a Logistic Regression model (see [de Villiers and Brümmer, 2010]) is performed on 2008 data to learn scaling and offset parameters for each condition separately. The training optimizes cross-entropy with a given target prior. The same target speaker prior is used for the calibration as used in the official DCF measures in Chapter 2.1. As the new DCF measure did not appear as a meaningful measure during the experiments, a target prior $P_{\text{Target}} = 0.1$ is used to focus the calibration on an area around the old DCF point. Such objective function calibrates the scores (makes them interpretable as log-likelihood ratios) for a wide range of operating points around the chosen one. For a thorough theoretical investigation on calibration see [Brümmer, 2010].

To show and analyze these calibrations for different operating points on the development as well as on the test sets, normalized DCF plots (see [de Villiers and Brümmer, 2010] or [Brümmer et al., 2010]) are used. Figure 5.5.a shows a normalized DCF plot for the `SNERF+DCT-iV` system on the *tel-phn:tel-phn* conditions. The y-axis gives the normalized minimum and actual DCF measures which are plotted against operating points (parameterized by the target prior), on the x-axis. A good indicator for the quality of the calibration and how the system performs on new data, is how closely the actual DCF curve (thin red line) follows the minimum DCF curve (thick red line) for the evaluation data. For this condition, a good calibration and a good generalization to the test data is achieved. The actual DCF curve stays close to the optimum and does not drift apart towards the new DCF point (much lower target prior). At the old DCF point, a minimal value of 0.43 and an actual DCF of around 0.47 is achieved. Note, the new DCF point is parameterized at $\text{logit}P_{\text{tar}} = -6.9$ and is not plotted, as even the new min DCF is around 1.

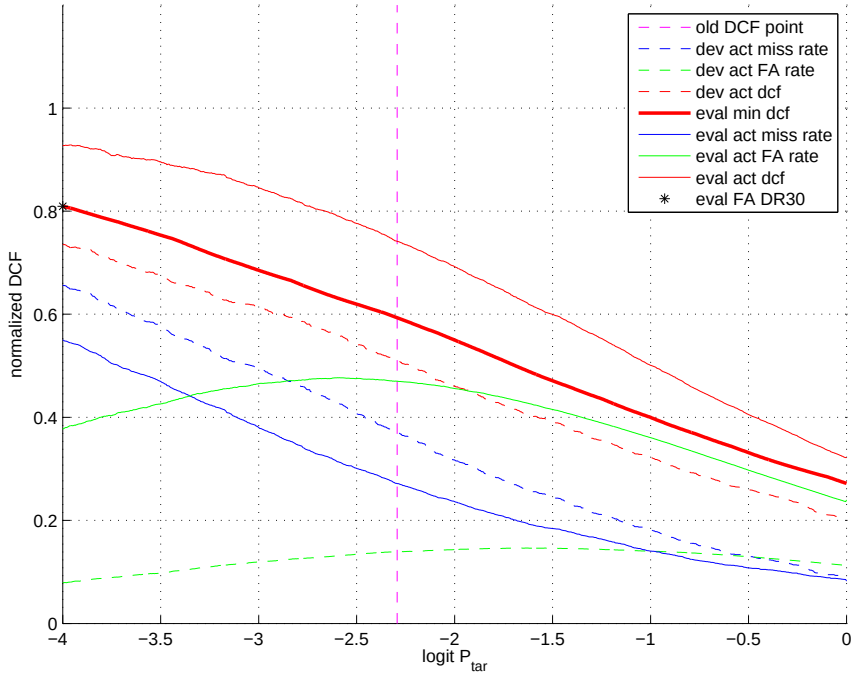
Figure 5.5.b shows the normalized DCF plot for the *tel-phn:int-mic* condition. It can be observed, that the calibration is much worse for this condition. It does not drift apart completely, but at the old DCF point, a minimum value of 0.59 and an actual value of around 0.75 is obtained.

Figure 5.6.a shows the normalized DCF plot for the *int-mic:int-mic* condition. The plot shows a very good calibration over the complete range of the target prior. At the old DCF point, the minimal and the actual value differ only slightly.

As mentioned, the systems are calibrated separately per condition. As the conditions are not treated separately during feature extraction or modeling, it would be desirable to obtain a well working condition-independent calibration (or no calibration at all, using the PLDA model). However, Figure 5.6.b shows that this is not the case. Here, the system is calibrated on the three merged conditions of the development set and the resulting offset and scaling factors are applied to the system outputs, independently of the condition. The plot exemplifies how the very good calibration on the matched condition (subplot a) is degraded a lot (subplot b), when the calibration parameters are estimated on unmatched and mixed conditions. This indicates that there is still a need to set a separate threshold

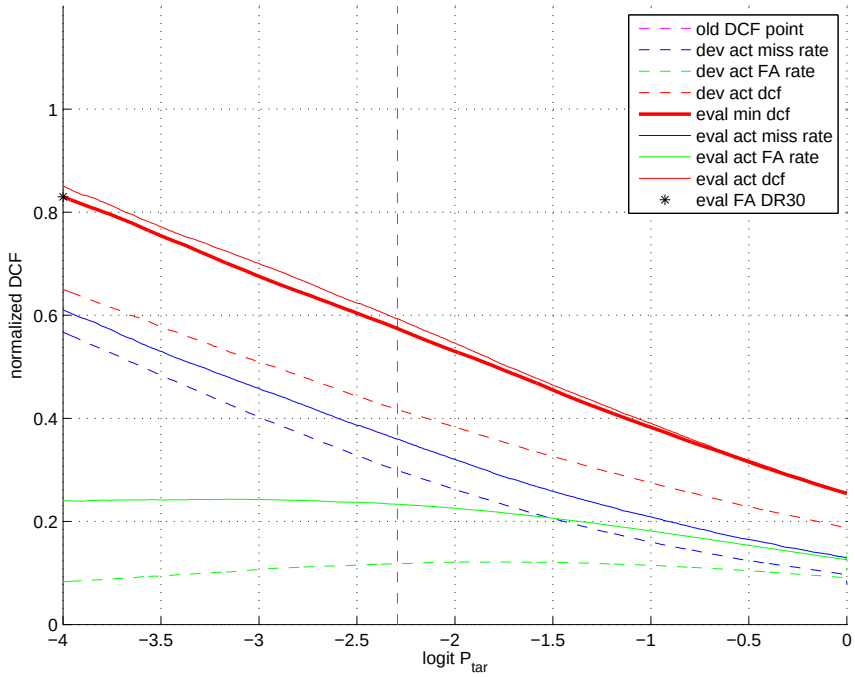


(a) *tel-phn:tel-phn*

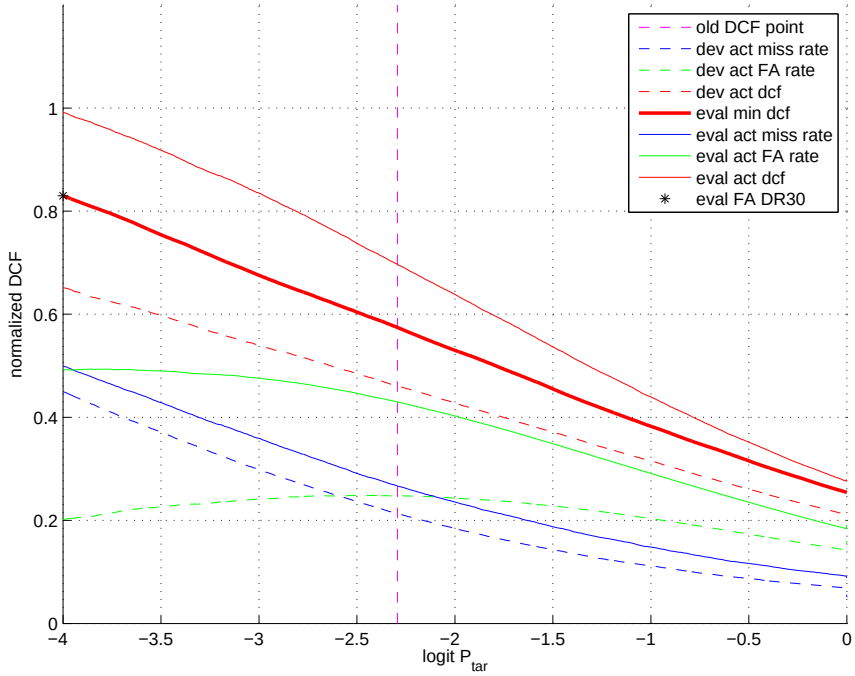


(b) *tel-phn:int-mic*

Figure 5.5: Normalized DCF plot for *tel-phn:tel-phn* and *tel-phn:int-mic* condition showing the quality of the calibration of `DCT+SNERF-iV` system when calibrated on the corresponding development set.



(a) Calibrated on *int-mic-int:mic*



(b) Calibrated on merged conditions

Figure 5.6: Normalized DCF plot for *int-mic:int-mic* condition showing the quality of the calibration of `DCT+SNERF-iV` system when calibrated on the corresponding development set and on the merged development set conditions.

System	DEV SRE 2008			EVAL SRE 2010		
	EER	oDCF	nDCF	EER	oDCF	nDCF
Cepstral iVector system `CEP-iV`	2.02	0.90	4.71	3.14	1.55	5.04
Concatenated `CEP+DCT-iV`	1.69	0.80	4.00	2.72	1.36	4.31
Concatenated `CEP+SNERF-iV`	**1.65**	0.80	3.89	2.74	1.34	4.44
Concatenated `CEP+DCT+SNERF-iV`	1.70	**0.75**	**3.68**	**2.63**	**1.29**	**4.21**
Score fusion `CEP-iV`, `DCT-iV` & `SNERF-iV`	1.92	0.78	4.06	3.09	1.49	4.47

Table 5.1: Results (old and new DCF $\times 10$) for single cepstral baseline system (`CEP-iV`) and for combinations with one or two prosodic iVector systems.

for different conditions.

This might also explain the generally worse calibration on the mixed *tel-phn:int-mic* condition. First, it was already observed that the system performance is generally worse, most probably due to lack of appropriate training data. But also, the *tel-phn:int-mic* condition is unmatched between training and test itself, with telephone speech for enrollment and microphone speech for test, making it probably harder to calibrate, as the score distribution is quite different from the matched conditions.

5.3 Combination with cepstral baseline system

The justification for the use of higher-level systems usually lies in an overall improvement by fusion with a cepstral baseline system. In this section, it is evaluated how a current state-of-the-art cepstral speaker verification system can be improved by combination with the prosodic systems developed during the work on this thesis. Not only results of classical fusions of systems on the score level are presented, but we also present a novel way of combining heterogeneous low-level cepstral and high-level prosodic systems on the iVector level, similar to the combination of the two iVector-based prosodic systems in Chapter 5.1.

5.3.1 *tel-phn:tel-phn* condition

The baseline system is a cepstral iVector PLDA system. This system is an improved version of the best-performing individual system from the Agnitio-Brno-CRIM NIST SRE 2010 submission [Brümmer et al., 2010]. It is based on 60-dimensional cepstral features and a 2048-component full covariance UBM. 400-dimensional iVectors are used and the dimension is further reduced to 200 by standard LDA, and the iVectors are further normalized to unity length before PLDA modeling. Note, that this pre-processing of iVectors is very helpful for cepstral iVectors, but did not show any significant improvement for the prosodic iVectors. The first row of Table 5.1 gives the results for the two data sets on the *tel-phn:tel-phn*

condition.

Again, the iVector nature of the baseline system allows us to use a novel way of combining low- and high-level systems by simple concatenation of their iVectors and joint PLDA modeling. Two prosodic iVectors, based on DCT and SNERF features, as presented in Section 5.1, are used. First, an LDA reduction to 200 dimensions is used and length-normalization is applied to both sets of prosodic iVectors. As mentioned above, this does not significantly affect the performance of the prosodic systems, but it is desirable to use the same pre-processing as used for the cepstral iVectors.

In this way, three same sized sets of 200 dimensional iVectors are available. One cepstral (`CEP-iV`) and two prosodic (`DCT-iV` and `SNERF-iV`). First, the combination of a single prosodic set with the cepstral set is investigated. The cepstral iVectors are concatenated separately with each of the prosodic iVector sets, to obtain two sets of four hundred-dimensional iVectors. Then, a standard PLDA model with full rank of 400 is trained for each type of combination. The second and third rows of Table 5.1 give the results for these combinations. A significant improvement for both *iVector fusions* of cepstral and prosodic features is achieved. Both fusions achieve about the same gains with up to 17% relative reduction on new DCF.

Finally, all three iVector types (one cepstral and two prosodic) are concatenated and a single PLDA model with full rank of 600 is trained. The fourth row of Table 5.1 gives the results for this combination. Further improvements are achieved, leading to reductions as high as 21% relative on the challenging new DCF measure.

As a last experiment, this approach is compared to the conventional score-level fusion. For this purpose, a Linear Logistic Regression [de Villiers and Brümmer, 2010] is trained to fuse the three individual system scores on the matched condition of the development set, and to apply the fusion parameters to the evaluation set. The last row of Table 5.1 indicates that consistent gains are also achieved by score-level fusion (as high as 13% relative on new DCF), but joint PLDA training of concatenated iVectors remains superior. iVector fusion of the cepstral system and the simple prosodic `DCT-iV` system already outperforms the score-level fusion of all three systems.

5.3.2 *tel-phn:int-mic* condition

Next, the behavior on the mixed *tel-phn:int-mic* condition is analyzed. The first row of Table 5.2 gives the results for the baseline system. As for the prosodic system, the performance is generally worse on this condition for the 2008 data set. Especially the EER and old DCF is about 50% worse than the measure obtained for the *tel-phn:tel-phn* condition in Table 5.1. Interestingly, this is not the case for the 2010 data. The cepstral iVector system achieves better results on this condition than on the tel-tel condition incorporating only telephone speech, which is not consistent with our prosodic results.

Again, a separate iVector combination of the cepstral baseline system with the two prosodic iVectors is performed. The second row of Table 5.2 shows consistent and significant gains due to the iVector fusion with the simple DCT features. Around 10% relative reduction in terms of the new DCF can be achieved, on both test sets and even higher

System	DEV SRE 2008			EVAL SRE 2010		
	EER	oDCF	nDCF	EER	oDCF	nDCF
Cepstral iVector system `CEP-iV`	3.78	1.90	5.23	2.95	1.32	4.26
Concatenated `CEP+DCT-iV`	**2.96**	1.48	4.76	**2.77**	**1.23**	**3.74**
Concatenated `CEP+SNERF-iV`	3.36	1.60	4.80	3.23	1.33	4.20
Concatenated `CEP+DCT+SNERF-iV`	3.21	**1.42**	**4.70**	3.30	1.36	3.81
Score fusion `CEP-iV`, `DCT-iV` & `SNERF-iV`	3.20	1.56	4.93	3.10	1.29	4.10

Table 5.2: Results (old and new DCF ×10) for *int-mic:tel-phn* condition for single cepstral baseline system (`CEP-iV`) and for combinations with two prosodic iVector systems.

gains for the 2008 EER and old DCF. The next row in the table gives the combination with the SNERF-based iVectors. Consistent gains are also obtained due to the iVector fusion, but these are smaller than for the DCT features. This behavior might be linked to generally worse performance of the `SNERF-iV` system on the second and third condition as reported in Section 5.1.

Similarly as for the telephone condition, the performance for the 600-dimensional iVectors incorporating one set of cepstral and two sets of prosodic features is investigated. The fourth row of Table 5.2 indicates that there is no such clear improvement due to the use of two diverse prosodic feature sets anymore. Only slight improvements on the old and new DCF measures of the development set can be achieved. For all other measures, it stays behind the simple `DCT-iV` combination. Also, the results for the score level fusion of all three systems are inconsistent on this condition. A direct comparison to the iVector fusion shows minor advantages on EER and 2010 old DCF. Still, the best results are obtained by iVector fusion of either two or three systems.

5.3.3 *int-mic:int-mic* condition

Finally, fusion results for the *int-mic:int-mic* conditions will be investigated. Again, the first row of Table 5.3 shows the results for the cepstral baseline system. Except for the EER and old DCF on 2008, it can be observed that the baseline system gives similar performance on all three conditions. The general picture for the fusion results is quite similar to the previous condition. Improvements are obtained for nearly all combinations of the baseline system with one or two prosodic systems. Again, the simple `DCT-iV` combination seems to be very robust and gives very good results. The combination with the `SNERF-iV` system does not work that well, especially on 2010 data.

Similarly as for the previous condition, it can be observed that the generally worse performance of the SNERF system on the third condition, as observed in Section 5.1, also seems to affect the combination with the baseline system.

System	DEV SRE 2008			EVAL SRE 2010		
	EER	oDCF	nDCF	EER	oDCF	nDCF
Cepstral iVector system `CEPiV`	5.16	1.96	4.92	3.4	1.61	4.83
Concatenated `CEP+DCT-iV`	4.08	1.53	**4.16**	**2.92**	**1.38**	**4.49**
Concatenated `CEP+SNERF-iV`	4.32	1.63	4.43	3.50	1.57	4.71
Concatenated `CEP+DCT+SNERF-iV`	**3.90**	**1.50**	4.24	3.5	1.55	4.71
Score fusion `CEP-iV`, `DCT-iV` & `SNERF-iV`	4.68	1.66	4.37	3.4	1.53	4.77

Table 5.3: Results (old and new DCF $\times 10$) for *int-mic:int-mic* condition for single cepstral baseline system (`CEP-iV`) and for combinations with two prosodic iVector systems.

5.3.4 Final observations

Resuming the fusion experiments (of low- and high-level systems) in terms of improvements achieved on the new DCF measure, which was introduced as the primary measure in the last NIST SRE evaluation, it can be said, that the highest relative reduction can be obtained for the *tel-phn:tel-phn* condition. 22% relative reduction on 2008 data and 17% on the extended 2010 data set were obtained. For the two conditions including microphone speech, also consistent but smaller improvements are achieved: for the *tel-phn:int-mic* condition, around 10% relative for the 2008 development set and 12% on the 2010 evaluation set. The relative reduction on new DCF for the *int-mic:int-mic* condition is 15% for 2008 data and around 7% for the 2010 test set. Interestingly, except for the purely telephone condition, often the highest gains are obtained from a combination with the simple DCT- contour based iVectors, without using the much more complex SNERFs.

6 Conclusions

6.1 Summary

During the work on this thesis, I could significantly reduce the error rates for prosodic systems for automatic speaker verification. Starting with over 20%, I could eventually quarter the EER on NIST 2008 data set and reach an EER around 5% on conversational English telephone tasks.

This was mainly due to improvements and new developments in terms of prosodic feature extraction as well as modeling techniques for prosodic features.

6.1.1 Extraction of prosodic contour features

In Chapter 3, I present the techniques for parameterization of speech which were investigated and developed during this thesis.

In the first period of the thesis, I worked on so called prosodic contour features, which model the continuous trajectories of pitch and energy extracted over long temporal contexts of speech, usually syllables. While my initial idea to use a parameterization of continuous pitch and energy trajectories for speaker verification was triggered by work from the speech synthesis area [Reichel, 2007], I found early that a similar approach of using a curve fitting algorithm on pitch and energy segments [Lin and Wang, 2005] was already successfully applied to prosodic speaker verification [Dehak et al., 2007]. However, my initial approach to model pitch and energy contours differed in the curve approximation as well as in the segmentation techniques. While the segmentation into syllable like units in [Dehak et al., 2007] is purely based on energy measures, I developed a technique to obtain pseudo-syllables from the output of a phone recognizer, as described in Section 3.1.2. This way, a segmentation technique was available, with complexity between the simple energy-based approach [Dehak et al., 2007] and language-dependent and LVCSR-based approach [Ferrer et al., 2007]. In the study presented in Section 3.1.5, I could show that the complexity of the segmentation highly correlates with the achieved system performance.

However, in the final experiments in Section 5.1, the best results are achieved with simple fixed-sized, long and highly overlapping windows. Due to the high overlap, many more feature frames per utterance can be extracted which indicates that a high amount of (obviously highly correlated) statistical evidence is more important than accurate segmentation

into correct linguistic units.

The curve approximation I propose, is simply based on discrete cosine transformation of the pitch and energy values, as described in Section 3.1.3, while in [Lin and Wang, 2005] it is proposed to use Legendre polynomials. In Section 3.1.5, I investigated both techniques and found both equivalent as they are both based on orthogonal basis functions and translate the variable length pitch or energy contour into a set of fixed-size de-correlated features. I also investigated the effect of voiced and unvoiced regions and found that it does not harm to just cut out the unvoiced regions (where no pitch is detected) prior to contour modeling. This way, there is no need for a special processing of undefined pitch values.

6.1.2 Modeling for prosodic contour features

Besides a different feature extraction process, I also started investigations into different modeling approaches. In the beginning, a simple Gaussian Mixture Modeling paradigm without any intersession compensation techniques was used as in Section 3.1.5. After small improvements using eigenchannel compensation for prosodic and cepstral contour features [Kockmann and Burget, 2008a], I could adopt the Joint Factor Analysis framework that was also proposed in [Dehak et al., 2007]. This modeling approach, incorporating speaker and channel compensation, was thoroughly analyzed theoretically and in experimental evaluations, as presented in Sections 4.3.1 and 4.3.4.

Significant improvements due to the JFA modeling with EERs around 15% could be achieved, similar to the results presented in [Dehak et al., 2007]. The DCT-based prosodic contour features in combination with JFA modeling were investigated as a prosodic system in the NIST SRE evaluation in 2008 [Burget et al., 2009a] and finally used in 2010 [Brümmer et al., 2010]. During this period, I investigated many aspects, such as amount and type of training data, the way of training the JFA model, different scoring techniques and modifications in the prosodic feature extraction as already mentioned, resulting in further improvements in EERs of around 10%.

Eventually, I could show in the final experiments, that the standard JFA modeling of prosodic contour features can be significantly outperformed by the iVector approach as presented in Section 4.3.2. Besides tests on conversational telephone speech, I could also show consistent gains of up to 20% relative due to the total variability modeling followed by PLDA on test conditions involving interview speech recorded over microphones. Note, that this improvement over JFA is observed only when the iVectors are modeled using the PLDA backend[1].

6.1.3 Modeling for SNERFs

The second phase of the thesis work was mainly focusing on the use of more complex prosodic features such as the Syllable-based Nonuniform Extract Region Features as described in Section 3.2. In a cooperative work with the STAR Lab at SRI International,

[1]No gain was observed during SRE 2010 system development [Brümmer et al., 2010] when iVectors were modeled with simpler scoring techniques [Dehak et al., 2007].

I focused on modeling techniques, making subspace modeling (incorporating speaker and session variability compensation) possible and working for heterogeneous high dimensional prosodic features such as the SNERFs.

SNERFs contain many diverse measurements of duration, pitch and energy that may further be undefined, as they are not only based on syllables, but also on the onset, nucleus or coda of a syllable. Also, they capture even a longer context of up to three syllables or pauses within two syllables. All these attributes and the fact that the raw SNERF features are very high-dimensional makes it hard to apply the JFA paradigm to these features. While SRI provided the SNERFs, I worked on developing a model that transfers the basic idea of subspaces for speaker and session variability modeling, from mean parameters of Gaussian Mixture Models to multinomial distributions.

The so-called Subspace Multinomial Model (SMM) is presented theoretically in Section 4.4.1, and in Section 4.4.2 it is shown how to apply it to SNERFs. Experimental results indicate, that it can outperform the standard form of Support Vector Machine modeling for SNERFs as proposed in [Ferrer et al., 2007]. Following this, I could further improve its performance by PLDA modeling as presented in Section 4.4.2. With an EER of 6.9% on 2008 data, the best results for a single prosodic system to date could be achieved, greatly outperforming the JFA modeling technique of contour features, which had been the most popular prosodic system till then.

However, the results presented in [Kockmann et al., 2011c] and the final experiments in Section 5.1 indicate, that the modeling of GMM based iVectors for DCT contour features is similar, if not better or more robust, than the iVector modeling of counts of complex SNERFs. Especially for conditions involving interview microphone speech, more consistent results are obtained for the approach modeling GMM based iVectors based on simple DCT contour features.

6.1.4 iVector fusion

Furthermore, I could present a novel way of combining the two best performing prosodic systems, one where iVectors based on GMMs are used to model simple DCT features extracted from uniform regions, and another where iVectors based on multinomial distributions are used to model a complex set of syllable-level features. These two systems are different at both, the feature and the modeling levels. Gains in the order of 20% are shown when combining these two systems with respect to the single best, resulting in an EER of 5.4% on 2008 data.

The combination is performed using iVector-level fusion: the individual iVectors for the two systems are concatenated and the joint iVectors are modeled using a single PLDA model. An important advantage of iVector-level fusion compared to score-level fusion is, that it can make use of the full information encoded in the iVectors, while for the score-level fusion, all information is already reduced to a single number per system. Besides conversational telephone speech, consistent gains are also presented for conditions involving mixed telephone and microphone and pure interview microphone conditions in the comparative study in Section 5.1.

The iVector-level fusion technique followed by PLDA modeling can also be applied to fuse highly heterogeneous features, such as low-level cepstral and high-level prosodic features. Using this procedure, I could achieve 20% relative improvement on new DCF over a cepstral iVector baseline, significantly outperforming score-level fusion. These are, to my knowledge, the largest relative gains obtained in speaker verification from combination of cepstral systems with prosodic features in several years.

6.2 Current state and future work

Since the NIST 2008 evaluation for speaker verification systems, the use of prosodic information to enhance acoustic state-of-the-art systems has become very popular again [Kajarekar, 2009, Kenny et al., 2008a, Yan, 2008]. This was mainly due to the quite simple but effective system originally proposed in [Dehak et al., 2007]. This framework was further investigated and enhanced during this thesis, but also by other sites.

While in [Ferrer et al., 2010], large gains in performance could be obtained by increasing the complexity of the system, I could obtain significant improvements through a further simplification, by the use of fixed length long temporal windows. In combination with the proposed iVector modeling by PLDA, I believe that this framework could become a standard for prosodic modeling. The prosodic feature extraction is simple and only needs two measurements from an audio signal – pitch and energy. The framework for iVector modeling of mean parameters and the Probabilistic Linear Discriminant Analysis are currently also the best techniques to model cepstral features [Kenny, 2010, Burget et al., 2011, Brümmer et al., 2010], so that they are already integrated in many speaker verification systems. Furthermore, I could show that the cepstral and the prosodic iVector frontends can be elegantly and efficiently combined by iVector fusion.

However, for the proposed Subspace Multinomial Modeling of SNERFs, I still see a lot of future work to be made.

6.2.1 Prosodic feature extraction

The features are still too complex to compute to be broadly adopted. Furthermore, they are language-dependent and are based on the output of an LVCSR system. In an ongoing cooperative work, we will present results for the subspace model on new language-independent set of SNERFs developed at SRI. Preliminary results show only a minor degradation compared to the language-dependent features. Still, the features are based on the output of a multilingual speech recognition system. Simpler segmentation techniques, based on pseudo-syllables, or even fixed-length windows might be considered. Also, the parameterization by GMMs might be optimized. One might think of using Variational Bayes methods to automatically determine the number of Gaussian components per SNERF token.

6.2.2 Prosodic modeling

In my opinion, most efforts should be devoted here. The experiments often show a rapid decrease in performance towards the very low false alarm regions. Currently, this behavior can not be explained and further investigations need to be made. Further, inconsistent behavior is observed especially on test conditions incorporating speech styles and channels other than conversational telephone speech.

One major concern is to improve the model training of the SMM. As described in Section 4.4, an iterative optimization scheme is used for the model parameter estimates, to merely maximize the likelihood of the training data. Instead, it would be preferable to use a probabilistic model that also imposes prior information on the model parameters, similar to the EM algorithm developed for the JFA model modeling distributions of GMM mean supervectors (see Section 4.3.1). However, it is not easy to impose a prior on the latent vector and to calculate its posterior distribution for the likelihood function incorporated in the Subspace Multinomial Model, which is based on a nonlinear softmax function. A possible solution might be found in the literature for Bayesian solutions of Multiclass Logistic Regression parameter estimation. In [Bouchard, 2007], different methods are compared and a Variational Bayes approach is presented which approximates the softmax function by a product of logistic sigmoids, adapting a solution for Logistic Regression [Jaakkola and Jordan, 2000].

An interesting finding is, that the `DCT-iV` system, which models the mean parameters of simple DCT features, gives similar error rates to the much more complex count modeling of SNERFs. This indicates, that the standard GMM based iVectors are still more powerful under the subspace paradigm than SMM modeling of counts of discrete classes defined by Gaussian mixture components. This seems reasonable and raises the question, if it is also desirable to find a way to appropriately model the GMM mean parameters for the SNERF features. Theoretically, this should be doable with the EM algorithm used for SNERF parameterization [Kajarekar et al., 2004].

Furthermore, directly combining subspace training for multinomial distributions and mean parameters of Gaussian distributions for a single set of features might be even more effective. A similar approach is used in the SGMM paradigm [Povey and Burget, 2011] used for automatic speech recognition.

Bibliography

[Abramowitz and Stegun, 1972] Abramowitz, M. and Stegun, I. A., editors (1972). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, chapter 8: Legendre Functions, pages 331–339.

[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.

[Atal, 1972] Atal, B. S. (1972). Automatic speaker recognition based on pitch contours. *J. Acoust. Soc. Am.*, 52:1687–1697.

[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.

[Bartkova et al., 2002] Bartkova, K., Gac, D. L., Charlet, D., and Jouvet, D. (2002). Prosodic parameter for speaker identification. In *Proc. of ICSLP, Denver, USA*, pages 1197–1200.

[Bishop, 2006] Bishop, C. (2006). *Pattern recognition and machine learning*.

[Bouchard, 2007] Bouchard, G. (2007). Efficient bounds for the softmax and applications to approximate inference in hybrid models. In *Proc. of NIPS 2007 Neural Information Processing Systems Conference, Whistler, Canada*.

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

[Brümmer, 2010] Brümmer, N. (2010). EM for Probabilistic LDA. Unpublished.

[Brümmer, 2010] Brümmer, N. (2010). *Measuring, refining and calibrating speaker and language information extracted from speech*. PhD thesis, Stellenbosch University.

[Brümmer et al., 2010] Brümmer, N., Burget, L., Kenny, P., Matějka, P., de Villiers, E., Karafiát, M., Kockmann, M., Glembek, O., Plchot, O., Baum, D., and Senoussauoi, M. (2010). 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 speaker partitioning problem. In *Proc. of Odyssey, Brno, Czech Republic*.

[Brümmer and de Villiers, 2011] Brümmer, N. and de Villiers, E. (2011). The BOSARIS toolkit user guide: Theory, algorithms and code for binary classifier score processing. Documentation of BOSARIS toolkit.

[Burget et al., 2008] Burget, L., Brümmer, 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. Technicla Report from JHU Summer Workshop '08.

[Burget et al., 2009a] Burget, L., Fapšo, M., Hubeika, V., Glembek, O., Karafiát, M., Kockmann, M., Matějka, P., Schwarz, P., and Černocký, J. (2009a). BUT system for NIST 2008 speaker recognition evaluation. pages 2335–2338.

[Burget et al., 2009b] Burget, L., Matějka, P., Hubeika, V., and Černocký, J. (2009b). Investigation into variants of Joint Factor Analysis for speaker recognition. *Interspeech, Brighton, UK*, pages 1263–1266.

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

[Burget et al., 2011] Burget, L., Plchot, O., Cumani, S., Glemebk, O., Matějka, P., and Brümmer, N. (2011). Discriminatively trained Probabilistic Linear Discriminant Analysis for speaker verification. In *Proc. of ICASSP, Prague, Czech Republic*.

[Carey et al., 1996] Carey, M., Parris, E., Lloyd-Thomas, H., and Bennett, S. (1996). Robust prosodic features for speaker identification. In *Proc. of Fourth International Conference on Spoken Language, Philadelphia*, pages 1800–1803.

[Chang and Lin, 2001] Chang, C. and Lin, C. (2001). Libsvm : a library for support vector machines. `http://www.csie.ntu.edu.tw/~libsvm`.

[Davis and Mermelstein, 1980] Davis, S. 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(pp. 1–4):pp. 357–366.

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

[Dehak et al., 2009a] Dehak, N., Dehak, R., Kenny, P., and Brümmer, N. (2009a). Support vector machines versus fast scoring in the low-dimensional total variability space.

[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., 2009b] Dehak, N., Kenny, P., eda Dehak, R., Dumouchel, P., and Ouellet, P. (2009b). Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech and Language processing*, pages pp. 1–23.

[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.

[Fant et al., 1990] Fant, G., Kruckenberg, A., and Nord, L. (1990). Prosodic and segmental speaker variations. In *Proc. of Workshop on Speaker Characterization in Speech Technology*, pages 106–120.

[Ferrer, 2009] Ferrer, L. (2009). *Statistical Modeling of Heterogeneous Features for Speech Processing Tasks*. PhD thesis, Stanford University.

[Ferrer et al., 2010] Ferrer, L., Scheffer, N., and Shriberg, E. (2010). A comparison of approaches for modeling prosodic features in speaker recognition. *Proc. of ICASSP, Dallas*.

[Ferrer et al., 2007] Ferrer, L., Shriberg, E., Kajarekar, S., and Sönmez, K. (2007). Parameterization of prosodic feature distributions for SVM modeling in speaker recognition. *Proc. ICASSP, Taipei*, 4:233–236.

[Fletcher, 2000] Fletcher, R. (2000). *Practical Methods of Optimization*. Wiley, second edition.

[Fukunaga, 1990] Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition*. Academic Press, second edition.

[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.

[Gauvain and Lee, 1994] Gauvain, J.-L. 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, 2009] Glembek, O. (2009). Joint Factor Analysis Matlab demo. `http://speech.fit.vutbr.cz/en/software/joint-factor-analysis-matlab-demo`.

[Glembek et al., 2009] Glembek, O., Burget, L., Dehak, N., Brümmer, N., and Kenny, P. (2009). Comparison of scoring methods used in speaker recognition with joint factor analysis. *Proc. of ICASSP, Taipei*.

[Glembek et al., 2008] Glembek, O., Matějka, P., Burget, L., and Mikolov, T. (2008). Advances in phonotactic language recognition. In *Proc. Interspeech, Brisbane, Australia*, pages 743–746.

[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.

[Jaakkola and Jordan, 2000] Jaakkola, T. and Jordan, M. (2000). Bayesian parameter estimation via variational methods. pages 25–37.

[Jolliffe, 2002] Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer, second edition.

[Jorge and Stephen, 2006] Jorge, N. and Stephen, W. (2006). *Numerical Optimization*. Springer, second edition.

[Kajarekar et al., 2004] Kajarekar, S., Ferrer, L., Sönmez, K., Zheng, J., Shriberg, E., and Stolcke, A. (2004). Modeling NERFs for speaker recognition. In *Proc. Odyssey, Toledo, Spain*, pages 51–56.

[Kajarekar et al., 2003] Kajarekar, S., Ferrer, L., Venkataraman, A., Sönmez, K., Shriberg, E., Stolcke, A., and Gadde, R. R. (2003). Speaker recognition using prosodic and lexical features. In *Proc. of IEEE Speech Recognition and Understanding Workshop (ASRU)*.

[Kajarekar, 2009] Kajarekar, S. e. a. (2009). The SRI NIST 2008 speaker recognition evaluation system. In *Proc. of ICASSP '09, Taipei*, pages 4205–4208.

[Kenny, 2006] Kenny, P. (2006). Joint Factor Analysis of speaker and session variability: Theory and algorithms. Technical report. `http://www.crim.ca/perso/patrick.kenny`.

[Kenny, 2010] Kenny, P. (2010). Bayesian speaker verification with heavy tailed priors. In *Keynote presentation, Odyssey, Brno, Czech Republic*.

[Kenny et al., 2005] Kenny, P., Boulianne, G., Ouellet, P., and Dumouchel, P. (2005). Factor analysis simplified. In *Proc. of the 2005 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2005), Philadelphia, PA, USA*, pages 637–640.

[Kenny and Dumouchel, 2004] Kenny, P. and Dumouchel, P. (2004). Disentangling speaker and channel effects in speaker verification. In *Proc. of ICASSP*, pages 37–40.

[Kenny et al., 2003] Kenny, P., Mihoubi, M., and Dumouchel, P. (2003). New MAP estimates for speaker recognition. In *Proc. Eurospeech, Geneva, Switzerland*.

[Kenny et al., 2008a] Kenny, P., N, D., and Ouellet, P. (2008a). The CRIM systems for the NIST 2008 speaker recognition evaluation. In *Proc. of 2008 NIST Speaker Recognition Evaluation Workshop, Montreal, CA*.

[Kenny et al., 2008b] Kenny, P., Ouellet, P., Dehak, N., Gupta, V., and Dumouchel, P. (2008b). A study of inter-speaker variability in speaker verification. *IEEE Trans. Audio*.

[Kinnunen and Li, 2010] Kinnunen, T. and Li, H. (2010). An overview of text-independent speaker recognition: From features to supervectors. *Speech Communication*, 52(1):pp. 12–40.

[Kockmann, 2006] Kockmann, M. (2006). Nutzung phonetischer Merkmale zur Sprechererkennung. Master's thesis, Ilmenau, Technische Universität.

[Kockmann and Burget, 2008a] Kockmann, M. and Burget, L. (2008a). Contour modeling of prosodic and acoustic features for speaker recognition. pages pp. 45–48.

[Kockmann and Burget, 2008b] Kockmann, M. and Burget, L. (2008b). Syllable based feature-contours for speaker recognition. In *Proc. 14th International Workshop on Advances in Speech Technology*, page 4.

[Kockmann et al., 2010a] Kockmann, M., Burget, L., Glembek, O., Ferrer, L., and Černocký, J. (2010a). Prosodic speaker verification using Subspace Multinomial Models with intersession compensation. In *Proc. of Interspeech, Tokyo, Japan*.

[Kockmann et al., 2009] Kockmann, M., Burget, L., and Černocký, J. (2009). Brno University of Technology system for Interspeech 2009 Emotion Challenge. *Proc. Interspeech, Brighton*, pages pp. 348–351.

[Kockmann et al., 2010b] Kockmann, M., Burget, L., and Černocký, J. (2010b). Brno University of Technology system for Interspeech 2010 Paralinguistic Challenge. In *Proc. Interspeech 2010, Makuhari, Chiba, JP*, number 9, pages 2822–2825.

[Kockmann et al., 2010c] Kockmann, M., Burget, L., and Černocký, J. (2010c). Investigations into prosodic syllable contour features for speaker recognition. *Proc. of ICASSP, Dallas, USA*.

[Kockmann et al., 2011a] Kockmann, M., Burget, L., and Černocký, J. (2011a). Application of speaker- and language identification state-of-the-art techniques for emotion recognition. *Speech Communication*, 53(7):14.

[Kockmann et al., 2011b] Kockmann, M., Ferrer, L., Burget, L., Shriberg, E., and Černocký, J. (2011b). Recent progress in prosodic speaker verification. In *Proc. of ICASSP, Prague, Czech Republic*.

[Kockmann et al., 2011c] Kockmann, M., Ferrer, L., Burget, L., and Černocký, J. (2011c). iVector fusion of prosodic and cepstral features for speaker verification. In *Proc. of Interspeech, Florence, Italy*.

[Lin and Wang, 2005] Lin, C.-Y. and Wang, H.-C. (2005). Language identification using pitch contour information. *Proc. of ICASSP 2005, Philadelphia, PA*, pages 601–604.

[Ma et al., 2011] Ma, Y., Niyogi, P., Sapiro, G., and Vidal, R. (2011). Dimensionality reduction via subspace and submanifold learning. *IEEE Signal Processing Magazine*, 28(2):14–15.

[Martin and Przybocki, 2004] Martin, A. and Przybocki, M. (2004). 2004 NIST speaker recognition evaluation. `http://www.itl.nist.gov/iad/mig//tests/sre/2004`.

[NIST, 1996] NIST (1996). Speaker recognition evaluations website. `http://www.itl.nist.gov/iad/mig/tests/sre/`.

[NIST, 2005] NIST (2005). The NIST year 2005 speaker recognition evaluation plan.

[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`.

[Nolan, 1983] Nolan, F. (1983). *The Phonetic Bases of Speaker Recognition*. Cambridge Press.

[Pelecanos and Sridharan, 2001] Pelecanos, J. and Sridharan, S. (2001). Feature warping for robust speaker verification. In *Proc. of A Speaker Odyssey, Crete, Greece*.

[Peskin et al., 2003] Peskin, B., Navratil, J., Abramson, J., Jones, D., Klusacek, D., Reynolds, D., and Xiang, B. (2003). Using prosodic and conversational features for high-performance speaker recognition: report from JHU WS'02. In *Proc. of ICASSP, Hong Kong, China*, pages 792–795.

[Povey and Burget, 2011] Povey, D. and Burget, L. (2011). The Subspace Gaussian Mixture model – a structured model for speech recognition. *Computer Speech and Language*, 25:404–439.

[Prince, 2007] Prince, S. J. D. (2007). Probabilistic Linear Discriminant Analysis for Inferences About Identity. In *Proc. of ICCV*.

[Reichel, 2007] Reichel, U. (2007). Data-driven extraction of intonation contour classes. In *Proc. 6th ISCA Workshop on Speech Synthesis, Bonn*, pages 240–245.

[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 *Proc. of ICASSP, Hong Kong, China*.

[Reynolds et al., 2000] Reynolds, D., Quatieri, T., and Dunn, R. (2000). Speaker verification using adapted Gaussian Mixture Models. *Digital Signal Processing*, 10(1-3):pp. 19–41.

[Reynolds and Rose, 1995] Reynolds, D. A. and Rose, R. C. (1995). Robust text-independent speaker identification using Gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, 3:72–83.

[Scheffer et al., 2010] Scheffer, N., Ferrer, L., Graciarena, M., Kajarekar, S., Shriberg, E., and Stolcke, A. (2010). The SRI NIST 2010 speaker recognition evaluation system. *Proc. of ICASSP, Prague*.

[Schwarz, 2009] Schwarz, P. (2009). *Phoneme recognition based on long temporal context*. PhD thesis, Faculty of Information Technology, BUT.

[Schwarz et al., 2008] Schwarz, P., Matějka, P., Burget, L., and Glembek, O. (2008). Phoneme recognizer based on long temporal context. `http://speech.fit.vutbr.cz/en/software/`.

[Schwarz et al., 2006] Schwarz, P., Matějka, P., and Černocký, J. (2006). Hierarchical Structures of Neural Networks for Phoneme Recognition. *Proceedings of ICASSP 2006, Toulouse, France*, pages pp. 325–328.

[Shriberg et al., 2005] Shriberg, E., Ferrer, L., Kajarekar, S., and Venkataraman, A. (2005). Modeling prosodic feature sequences for speaker recognition. *Speech Communication*, 46(3–4):455–472.

[Sönmez et al., 1998] Sönmez, K., Shriberg, E., Heck, L., and Weintrau, M. (1998). Modeling dynamic prosodic variation for speaker verification. In *Proc. of ICSLP, Sydney, Australia*, pages 3189–3192.

[Sönmez et al., 1997] Sönmez, M. K., Heck, L., Weintraub, M., and Shriberg, E. (1997). A lognormal tied mixture model of pitch for prosody-based speaker recognition. In *Proc. of Eurospeech, Rhodes, Greece*, pages 1391–1394.

[Soufifar et al., 2011] Soufifar, M., Kockmann, M., Burget, L., Plchot, O., Glembek, O., and Svendsen, T. (2011). iVector approach to phonotactic language recognition. In *Proc. of Interspeech, Florence, Italy*.

[Talkin, 1995] Talkin, D. (1995). *Speech Coding and Synthesis*, chapter 14: A Robust Algorithm for Pitch Tracking (RAPT). Elsevier.

[Teunen et al., 2000] Teunen, R., Shahshahani, B., and Heck, L. (2000). A model based transformational approach to robust speaker recognition. In *Proc. of ICSLP*.

[Titze, 2000] Titze, I. R. (2000). *Principles of voice production*. National Center for Voice and Speech.

[Weber et al., 2002] Weber, F., Manganaro, L., Peskin, B., and Shriberg, E. (2002). Using prosodic and lexical information for speaker identification. In *Proc. of ICASSP, Orlando, USA*, pages 141–144.

[Yan, 2008] Yan, Y. e. a. (2008). Description of IOA systems for SRE08 - speaker recognition evaluation. In *2008 NIST Speaker Recognition Evaluation Workshop, Montreal, CA*.

A

Derivation of a Joint Factor Analysis Model

A.1 Likelihood of data for a GMM model

The log-likelihood of the data matrix $\mathbf{O}$ for a Gaussian mixture model is given by

$$\log p(\mathbf{O}) = \sum_t \log p(\mathbf{o}_t) = \sum_t \log \sum_c p(\mathbf{o}_t|c)p(c) \tag{A.1}$$

with Gaussians c and the sum over t running over all frames in $\mathbf{O}$. When training the GMM, we want to estimate the model parameters in such a way, that they maximize the likelihood of the data.

Rewriting A.1 we get

$$\log p(\mathbf{O}) = \sum_t \underbrace{\sum_c p(c|\mathbf{o}_t)}_{1} \log p(\mathbf{o}_t), \tag{A.2}$$

where by adding the sum over the posterior probabilities of Gaussian c given frame $\mathbf{o}_t$ we did not change the equation. Using Bayes' rule we obtain

$$\log p(\mathbf{O}) = \sum_t \sum_c p(c|\mathbf{o}_t) \log \frac{p(\mathbf{o}_t|c)p(c)}{p(c|\mathbf{o}_t)} \tag{A.3}$$

$$= \sum_t \sum_c p(c|\mathbf{o}_t) \log(p(\mathbf{o}_t|c)p(c)) - \sum_t \sum_c p(c|\mathbf{o}_t) \log p(c|\mathbf{o}_t) \tag{A.4}$$

$$= \sum_t \sum_c p(c|\mathbf{o}_t) \log p(\mathbf{o}_t|c) + \sum_t \sum_c p(c|\mathbf{o}_t) \log \frac{p(c)}{p(c|\mathbf{o}_t)}. \tag{A.5}$$

Assuming fixed assignment of frames to Gaussians, e.g. by the UBM

$$\gamma_c(t) = p(c|\mathbf{o}_t) = \frac{\pi_c \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}{\sum_{j=1}^{C} \pi_c \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)} \tag{A.6}$$

the right term of A.5 becomes constant w.r.t. the model parameters and we get our function to maximize:

$$\log p(\mathbf{O}) = \sum_t \sum_c \gamma_c(t) \log p(\mathbf{o}_t|c) + \text{const.} \tag{A.7}$$

Taking a Gaussian distribution

$$\mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) = p(\mathbf{o}_t|c) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_c|} \exp(-\frac{1}{2}(\mathbf{o}_t - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1}(\mathbf{o}_t - \boldsymbol{\mu}_c)), \quad \text{A.8}$$

and inserting the logarithm

$$\log p(\mathbf{o}_t|c) = -\frac{D}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_c| - \frac{1}{2}(\mathbf{o}_t - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1}(\mathbf{o}_t - \boldsymbol{\mu}_c) \quad \text{A.9}$$

into auxiliary function A.7, we obtain:

$$\log p(\mathbf{o}) = \sum_t \sum_c \gamma_c(t)[-\frac{D}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_c| - \frac{1}{2}(\mathbf{o}_t - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1}(\mathbf{o}_t - \boldsymbol{\mu}_c)] + \text{const} \quad \text{A.10}$$

A.2 Likelihood of data for a JFA model

For a Joint Factor Analysis model, the speaker-and utterance-dependent mean is

$$\boldsymbol{\mu}_c = \mathbf{m}_c + \mathbf{V}_c\mathbf{y}_s + \mathbf{U}_c\mathbf{x}_u + \mathbf{D}_c\mathbf{z}_s \quad \text{A.11}$$

with $\mathbf{m}$ being the speaker independent mean vector (often obtained from UBM, but can be retrained too), $\mathbf{V}$ and $\mathbf{U}$ the low-rank eigenvoice and eigenchannel matrices and vectors $\mathbf{y}_s$ and $\mathbf{x}_u$ the hidden variables. $\mathbf{D}$ is a diagonal matrix covering the residual speaker variability and $\mathbf{z}_s$ is the corresponding latent variable, which we will not use and set to zero, as it is not used in our experiments.

Inserting A.11 into A.10 and moving the first term in square brackets of A.10 also to the constant, we get

$$\begin{aligned}\log p(\mathbf{O}) = \sum_t \sum_c \gamma_c(t)[&-\frac{1}{2}\log|\boldsymbol{\Sigma}_c| \\ &- \frac{1}{2}(\mathbf{o}_t - (\mathbf{m}_c + \mathbf{V}_c\mathbf{y}_s + \mathbf{U}_c\mathbf{x}_u))^T \boldsymbol{\Sigma}_c^{-1}(\mathbf{o}_t - (\mathbf{m}_c + \mathbf{V}_c\mathbf{y}_s + \mathbf{U}_c\mathbf{x}_u))] + \text{const.}\end{aligned} \quad \text{A.12}$$

$$\begin{aligned}= \sum_t \sum_c \gamma_c(t)[&-\frac{1}{2}\log|u\boldsymbol{\Sigma}_c| - \frac{1}{2}(\mathbf{o}_t^T \boldsymbol{\Sigma}_c^{-1}\mathbf{o}_t - 2\mathbf{o}_t^T \boldsymbol{\Sigma}_c^{-1}\mathbf{m}_c - 2\mathbf{o}_t^T \boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s \\ &- 2\mathbf{o}_t^T \boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{x}_u + \mathbf{m}_c^T \boldsymbol{\Sigma}_c^{-1}\mathbf{m}_c + 2\mathbf{m}_c^T \boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\mathbf{m}_c^T \boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{x}_x + (\mathbf{V}_c\mathbf{y}_s)^T \boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s \\ &+ 2(\mathbf{U}_c\mathbf{x}_u)^T \boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s) + (\mathbf{U}_c\mathbf{x}_u)^T \boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{x}_u)] + \text{const.}\end{aligned} \quad \text{A.13}$$

Let's define statistics

$$\gamma_c = \sum_t \gamma_c(t) \quad \text{A.14}$$

$$\boldsymbol{\theta}_c = \sum_t \gamma_c(t)\mathbf{o}_t \quad \text{A.15}$$

where γ_c are the zero order statistics and $\boldsymbol{\theta}_c$ the first order statistics.

Changing the order of the sums and making use of A.14 and A.15, we can rewrite A.13 as

$$\sum_c [-\frac{1}{2}\gamma_c \log|\boldsymbol{\Sigma}_c| - \frac{1}{2}(\mathrm{tr}(\boldsymbol{\theta}_c^{2^T}\boldsymbol{\Sigma}_c^{-1}) - 2\boldsymbol{\theta}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{m}_c - 2\boldsymbol{\theta}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s - 2\boldsymbol{\theta}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{x}_u + \gamma_c\mathbf{m}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{m}_c + 2\gamma_c\mathbf{m}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\gamma_c\mathbf{m}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{x}_u + \gamma_c\mathbf{y}_s^T\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\gamma_c\mathbf{x}_u^T\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + \gamma_c\mathbf{x}_u^T\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c\mathbf{x}_u)] + \text{const.} \quad \text{A.16}$$

Here, we made use of the fact that the trace of a scalar is the same as the scalar, but we can re-arrange the matrices in the trace.

A.3 Posterior distribution of the hidden variable

We now want to derive the posterior distribution of a hidden variable, let us take $\mathbf{y}_s$ for speaker s. With Bayes rule we can write

$$p(\mathbf{y}_s|\mathbf{O}_s) = \frac{p(\mathbf{O}_s|\mathbf{y}_s)p(\mathbf{y}_s)}{p(\mathbf{O}_s)} \quad \text{A.17}$$

and

$$\log p(\mathbf{y}_s|\mathbf{O}_s) = \log p(\mathbf{O}_s|\mathbf{y}_s) + \log p(\mathbf{y}_s) + \text{const} \quad \text{A.18}$$

when taking the normalization part in the denominator as a constant.

Substituting A.16 into A.18 and moving all terms that do not depend on the hidden variable to the constant part, we get

$$\sum_u \sum_c [-\frac{1}{2}(-2\boldsymbol{\theta}_{uc}^T\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c\mathbf{y}_s + 2\gamma_{uc}\mathbf{m}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + \gamma_{uc}\mathbf{y}^T\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s)] + \log p(\mathbf{y}_s) + \text{const.} \quad \text{A.19}$$

Since we assume standard normal prior for the latent variable $\mathbf{y}_s$, from A.9 we obtain

$$\log p(\mathbf{y}_s|\mathbf{0},\mathbf{I}) = -\frac{D}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{I}| - \frac{1}{2}(\mathbf{y}_s-\mathbf{0})^T\mathbf{I}(\mathbf{y}_s-\mathbf{0}) = -\frac{1}{2}(\mathbf{y}_s^T\mathbf{y}_s) + \text{const}, \quad \text{A.20}$$

where $\mathbf{I}$ is the $D \times D$ identity matrix. Substituting this into A.19 and re-arranging gives

$$\sum_u -\frac{1}{2}(\mathbf{y}_s^T(\sum_c \gamma_{uc}\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c)\mathbf{y}_s) + \mathbf{y}_s^T\mathbf{I}\mathbf{y}_s - 2\sum_c \boldsymbol{\theta}_{uc}^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\sum_c \gamma_{uc}\mathbf{m}_c^T\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c\mathbf{y}_s + 2\sum_c \gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s) + \text{const} \quad \text{A.21}$$

$$= \sum_u -\frac{1}{2}(\mathbf{y}_s^T \underbrace{(\sum_c \gamma_{uc}\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c + \mathbf{I})}_{\mathbf{L}}\mathbf{y} - 2\underbrace{\sum_c(((\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T)\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c)\,\mathbf{y}_s)}_{\mathbf{b}}) + \text{const.} \tag{A.22}$$

By completing the square, it can be shown that the resulting quadratic form

$$-\frac{1}{2}(\mathbf{x}^T\mathbf{L}\mathbf{x} - 2\mathbf{b}\mathbf{x}) + \text{const} \tag{A.23}$$

corresponds to log of Gaussian distribution with mean $\boldsymbol{\mu} = \mathbf{b}\mathbf{L}^{-1}$ and covariance matrix $\mathbf{L}^{-1}$. So, the posterior distribution of the hidden variable $\mathbf{y}_s$ is Gaussian with

$$\mathcal{N}(\mathbf{y}_s|\sum_u\sum_c(\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T)\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{L}^{-1}, \mathbf{L}^{-1}). \tag{A.24}$$

Analogously, for $\mathbf{x}_u$ we will obtain

$$\mathcal{N}(\mathbf{x}_u|\sum_c((\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\mathbf{y}_s^T\mathbf{V}_c^T)\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c)\mathbf{L}^{-1}, \mathbf{L}^{-1}) \tag{A.25}$$

with

$$\mathbf{L} = \sum_c \gamma_c\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{U}_c + \mathbf{I}) \tag{A.26}$$

A.4 EM Estimation of low rank matrices V and U

Given utterances from many different speakers, we can estimate the posterior distribution of the hidden variables $\mathbf{y}_s$ for any speaker in the database as described in Section A.3. Keeping these distributions fixed, we can then re-estimate the low rank matrix itself.

Taking auxiliary function A.5 we obtain

$$\sum_s \int_{\mathbf{y}} q(\mathbf{y}_s)\log p(\mathbf{o}_s|\mathbf{y}_s)d\mathbf{y} + \text{const}, \tag{A.27}$$

where $q(\mathbf{y}_s) \propto p(\mathbf{y}_s|\mathbf{o}_s)$ is the posterior distribution of the latent variable given the current model estimates which are computed in the E-Step and kept fix. The sum over s runs over all speakers and `const` is again a constant part. Again, substituting A.16 into A.27 and moving all terms that do not depend on the matrix $\mathbf{V}$ to our constant part, we get

$$\sum_s \int_{\mathbf{y}} q(\mathbf{y}_s)\sum_u\sum_c -\frac{1}{2}[-2\boldsymbol{\theta}_{uc}^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\gamma_{uc}\mathbf{m}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + 2\gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s + \gamma_{uc}\mathbf{y}_s^T\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbf{y}_s]d\mathbf{y} + \text{const}. \tag{A.28}$$

Using first and second order expectations we rewrite A.28 to

$$\sum_s \sum_u \sum_c -\frac{1}{2}[-2\text{tr}(\mathbb{E}[\mathbf{y}_s]\boldsymbol{\theta}_{uc}^T \boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c) + 2\text{tr}(\mathbb{E}[\mathbf{y}_s]\gamma_{uc}\mathbf{m}_c^T\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c) + 2\text{tr}(\mathbb{E}[\mathbf{y}_s]\gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c) + \text{tr}(\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc}\mathbf{V}_c^T\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c)] + \text{const} \quad \text{(A.29)}$$

$$= \sum_s \sum_u \sum_c -\frac{1}{2}[-2\text{tr}(\mathbb{E}[y_s](\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T)\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c) + \text{tr}(\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc}\mathbf{V}_c^T\boldsymbol{\Sigma}_{uc}^{-1}\mathbf{V}_c)] + \text{const}, \quad \text{(A.30)}$$

where we have again made use of the "trace-trick" to re-arrange the matrices. Note, that the sum over u results from multiple utterances for speaker s, for which we also assume fixed estimates of $\mathbf{x}_u$ based on a fixed $\mathbf{U}$. Now we can take the derivative of A.30 with respect to $\mathbf{V}_c$, which is the part of the low-rank matrix corresponding to Gaussian c and set it equal to zero:

$$\frac{\partial}{\partial \mathbf{V}_c}\sum_s \sum_u -\frac{1}{2}[-2\text{tr}(\underbrace{\mathbb{E}[\mathbf{y}_s](\boldsymbol{\theta}_{uc}^T - \gamma_{uc}\mathbf{m}_c^T - \gamma_{uc}\mathbf{x}_u^T\mathbf{U}_c^T)\boldsymbol{\Sigma}_{uc}^{-1}}_{\mathbf{B}}\underbrace{\mathbf{V}_c}_{\mathbf{A}}) + \text{tr}(\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc}\mathbf{V}_c^T\boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c)] + \text{const}. \quad \text{(A.31)}$$

Using

$$\frac{\partial}{\partial \mathbf{A}} tr(\mathbf{AB}) = \mathbf{B}^T \quad \text{(A.32)}$$

we get

$$\sum_s \sum_u -\frac{1}{2}[-2\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{U}_c\mathbf{x}_u)\mathbb{E}[\mathbf{y}_s]^T) + \frac{\partial}{\partial \mathbf{V}_c}\text{tr}(\underbrace{\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc}}_{\mathbf{C}}\underbrace{\mathbf{V}_c^T}_{\mathbf{A}^T}\underbrace{\Sigma_c^{-1}}_{\mathbf{B}}\underbrace{\mathbf{V}_c}_{\mathbf{A}})] + \text{const}. \quad \text{(A.33)}$$

Using

$$\frac{\partial}{\partial \mathbf{A}}\text{tr}(\mathbf{CA}^T\mathbf{BA}) = \text{tr}(\mathbf{C}d\mathbf{A}^T\mathbf{BA}) + \text{tr}(\mathbf{CA}^T\mathbf{B}d\mathbf{A}) = \text{tr}(\mathbf{BAC}d\mathbf{A}^T) + \text{tr}(\mathbf{CA}^T\mathbf{B}d\mathbf{A}) = \mathbf{BAC} + (\mathbf{CA}^T\mathbf{B})^T, \quad \text{(A.34)}$$

we finally obtain

$$\sum_s \sum_u -\frac{1}{2}[-2\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{U}_c\mathbf{x}_u)\mathbb{E}[\mathbf{y}_s]^T) + \boldsymbol{\Sigma}_c^{-1}\mathbf{V}_c\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc}]. \quad \text{(A.35)}$$

Setting this to zero, multiplying from left by $\mathbf{\Sigma}_c$ and re-arranging, we get

$$\sum_s \sum_u [(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{U}_c\mathbf{x}_u)\mathbb{E}[\mathbf{y}_s]^T) - \mathbf{V}_c\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc}] = 0 \tag{A.36}$$

$$\Rightarrow \mathbf{V}_c = \sum_s \sum_u [(\boldsymbol{\theta}_c - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{U}_c\mathbf{x}_u)\mathbb{E}[\mathbf{y}_s]^T)(\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T]\gamma_{uc})^{-1}], \tag{A.37}$$

with $\mathbb{E}[\mathbf{y}_s] = \hat{\mathbf{y}}_s$ and $\mathbb{E}[\mathbf{y}_s\mathbf{y}_s^T] = \hat{\mathbf{y}}_s\hat{\mathbf{y}}_s^T + \mathbf{L}^{-1}$.

We can obtain an analogous solution for $\mathbf{U}_c$:

$$\mathbf{U}_c = \sum_s \sum_u [(\boldsymbol{\theta}_{uc} - \gamma_{uc}\mathbf{m}_c - \gamma_{uc}\mathbf{V}_c\mathbf{y}_s)\mathbb{E}[\mathbf{x}_u]^T)(\mathbb{E}[\mathbf{x}_u\mathbf{x}_u^T]\gamma_{uc})^{-1}], \tag{A.38}$$

assuming fixed $\mathbf{V}$.

B

Derivation of a Subspace Multinomial Model

In this section, we will derive an Iterative Reweighted Least Squares (IRLS) algorithm for the Subspace Multinomial Model proposed in Section 4.4. For simplicity we will derive the model for a single multinomial distribution.

The likelihood of data $\mathbf{O}$ for one utterance for a multinomial distribution can be computed as:

$$p(\mathbf{O}) = \prod_c p(c_c)^{\gamma_c} \tag{B.1}$$

with $p(c_c)$ being the probability of class $c = 1 \dots C$ and γ_c the zero order statistics (occupation counts) for class c. Taking the logarithm of it we get

$$\log p(\mathbf{O}) = \sum_c \gamma_c \log p(c_c) \tag{B.2}$$

which can be written as

$$\log p(\mathbf{O}) = \sum_c \gamma_c \log \frac{b(c_c)}{\sum_i b(c_i)} \tag{B.3}$$

where $b(c_c) \propto p(c_c)$. Now let us rewrite B.3 using a softmax function ϕ with factor analysis like parameters:

$$\log p(\mathbf{O}) = \sum_c \gamma_c \log \phi_c = \sum_c \gamma_c \log \frac{\exp(m_c + \mathbf{t}_c \mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i \mathbf{w})} \tag{B.4}$$

where m_c is a speaker-independent parameter for class c, $\mathbf{t}_c$ the row of a $C \times R_{\mathbf{T}}$ low-rank matrix $\mathbf{T}$ corresponding to class c and $\mathbf{w}$ the corresponding latent variable. $R_{\mathbf{T}}$ is the subspace size.

The maximum likelihood solution for re-estimation the the model parameters $\mathbf{T}$ and $\mathbf{w}$ does not lead to a closed form solution – as we have seen for the JFA model – due to the nonlinearities in the softmax function. However, an efficient iterative technique based on the Newton-Raphson iterative optimization scheme (see [Bishop, 2006, Chapter 4]) can be applied. It uses a local quadratic approximation to the log likelihood function.

The Newton-Raphson update for minimizing an error function $E(\mathbf{w}_n)$ (the negative log-likelihood) takes the form

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \mathbf{H}^{-1}\mathbf{g}, \tag{B.5}$$

where $\mathbf{g}$ is the gradient of the error function and $\mathbf{H}$ the Hessian matrix comprising the second derivatives of the error function with respect to $\mathbf{w}$.

First, we need the derivatives of the softmax function ϕ_c with respect to all of the parameters in $\mathbf{w}$. Applying the Quotient and Chain rule we obtain

$$\frac{\partial y_c}{\partial \mathbf{w}} = \frac{\mathbf{t}_c^T \exp(m_c + \mathbf{t}_c\mathbf{w}) \sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}) - \exp(m_c + \mathbf{t}_c\mathbf{w}) \sum_i \mathbf{t}_i^T \exp(m_i + \mathbf{t}_i\mathbf{w})}{(\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}))^2}. \quad \text{(B.6)}$$

This can be reformulated as:

$$\begin{aligned}
&\frac{\exp(m_c + \mathbf{t}_c\mathbf{w})(\mathbf{t}_c^T \sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}) - \sum_i \mathbf{t}_i^T \exp(m_c + \mathbf{t}_i\mathbf{w}))}{(\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}))^2} \\
&= \underbrace{\frac{\exp(m_c + \mathbf{t}_c\mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}}_{\phi_c} \left(\mathbf{t}_c^T \underbrace{\frac{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}}_{1} - \sum_i \mathbf{t}_i^T \underbrace{\frac{\exp(m_i + \mathbf{t}_i\mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}}_{\phi_i} \right) \\
&= \phi_c(\mathbf{t}_c^T - \sum_i \mathbf{t}_i^T \phi_i). \quad \text{(B.7)}
\end{aligned}$$

Using Eq. B.7 to derive the gradient of the error function (the negative of B.4) yields

$$\frac{\partial E}{\partial \mathbf{w}} = -\sum_c \gamma_c \frac{1}{\phi_c} \phi_c(\mathbf{t}_c^T - \sum_i \mathbf{t}_i^T \phi_i). \quad \text{(B.8)}$$

This can be simplified to

$$\begin{aligned}
-\sum_c \gamma_c \mathbf{t}_c^T - \sum_c \gamma_c \sum_i \mathbf{t}_i^T \phi_i &= -\sum_c \gamma_c \mathbf{t}_c^T - \sum_c \mathbf{t}_c^T \phi_c \sum_i \gamma_i \\
&= -\sum_c \gamma_c \mathbf{t}_c^T - \mathbf{t}_c^T \phi_c \sum_i \gamma_i = -\sum_c \mathbf{t}_c^T (\gamma_c - \phi_c \sum_i \gamma_i) \quad \text{(B.9)}
\end{aligned}$$

by changing the indices of the sums. Deriving it again yields to the Hessian matrix

$$\mathbf{H} = -\sum_c \mathbf{t}_c^T \phi_c (\mathbf{t}_c - \sum_i \mathbf{t}_i \phi_i) \sum_j \gamma_j. \quad \text{(B.10)}$$

Similarly, we can derive a Newton-Raphson update for the parameters of the subspace matrix $\mathbf{T}$ by minimizing the error function $E(\mathbf{t})$

$$\mathbf{t}^{new} = \mathbf{t}^{old} + \mathbf{H}^{-1}\mathbf{g}, \quad \text{(B.11)}$$

with $\mathbf{t}$ being an $R_\mathbf{T}C$ vector comprising the parameters $[\mathbf{t}_1 \dots \mathbf{t}_C]^T$, $\mathbf{g}$ being the $R_\mathbf{T}C$ dimensional gradient of the error function and $\mathbf{H}$ the $R_\mathbf{T}C \times R_\mathbf{T}C$ Hessian matrix comprising blocks of the second derivatives of the error function with respect to $\mathbf{t}$.

First, we need the derivatives of the softmax function ϕ_c with respect to all of the parameters in $\mathbf{t}_i$. Applying the Quotient and Chain rule we obtain

$$\frac{\partial \phi_c}{\partial \mathbf{t}_i} = \frac{\mathbf{w} \exp(m_c + \mathbf{t}_c\mathbf{w})\delta_{ci} \sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}) - \exp(m_c + \mathbf{t}_c\mathbf{w})\mathbf{w} \exp(m_i + \mathbf{t}_i\mathbf{w})}{(\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}))^2} \quad \text{(B.12)}$$

with δ_{ci} being the Kronecker-Delta function (equals one if $c = i$, zero otherwise). This can be reformulated as:

$$\begin{aligned}
&\frac{\mathbf{w} \exp(m_c + \mathbf{t}_c\mathbf{w})(\delta_{ci} \sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}) - \exp(m_i + \mathbf{t}_i\mathbf{w}))}{(\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w}))^2} \\
&= \mathbf{w} \underbrace{\frac{\exp(m_c + \mathbf{t}_c\mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}}_{\phi_c} \left(\delta_{ci} \underbrace{\frac{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}}_{1} - \underbrace{\frac{\exp(m_i + \mathbf{t}_i\mathbf{w})}{\sum_i \exp(m_i + \mathbf{t}_i\mathbf{w})}}_{\phi_i} \right) \\
&= \phi_c(\delta_{ci} - \phi_i)\mathbf{w}^T. \quad \text{(B.13)}
\end{aligned}$$

Using Eq. B.13 to derive the gradient of the error function (the negative of B.4) yields to

$$\frac{\partial E}{\partial \mathbf{t}_i} = -\sum_n \sum_c \gamma_{nc} \frac{1}{\phi_c} \phi_c(\delta_{ci} - \phi_i)\mathbf{w}_n^T = -\sum_n (\gamma_{ni} - \phi_i \sum_c \gamma_{nc})\mathbf{w}_n^T. \quad \text{(B.14)}$$

Deriving it again with respect to index j yields to the $D \times D$ block of the Hessian matrix

$$\mathbf{H}_{i,j} = \sum_n (\phi_j(\delta_{ij} - \phi_i) \sum_c \gamma_{nc})\mathbf{w}^T\mathbf{w}. \quad \text{(B.15)}$$