Brno University of Technology
Faculty of Information Technology
Department of Computer Graphics and Multimedia

# HABILITATION THESIS

## Subspace modeling techniques in speech processing

Lukáš Burget

September 2014

# Abstract

The recently introduced subspace modeling techniques revolutionized the field of speaker recognition. Dramatic performance improvements were observed in both speed and accuracy, which have increased the scale of viable speaker-id systems by several orders of magnitude. This habilitation thesis reviews the concept of channel compensation, which builds on the subspace modeling idea, and the concept of i-vectors representing sequences of continuous speech features by a low-dimensional fixed length vector. The main part of the thesis is an annotated collection of research publications covering various topics related to these subspace modeling techniques. Different variants of channel compensation techniques and i-vector models are analyzed and their applications to different speech processing problems such as speaker, language or emotion recognition are described. A publication that introduces the nowadays popular technique for i-vector based discriminative adaptation of a speech recognition system is also included. Conceptually new i-vector based approaches to fusion and to dicriminative training of speaker verification systems are proposed. Recent extensions and variations of the i-vector concept are discussed: Subspace n-gram model was introduced to model sequences of discrete features in prosodic speaker recognition or to represent phonotactics in language recognition. A simplified i-vector extraction model and its dicriminative training is proposed in order to facilitate implementation of i-vector into resource limited embedded devices. Finally, extensions of i-vector extractor robust to additive background noise are proposed in the included publications.

# Contents

# Chapter 1

# Introduction

The recently introduced subspace modeling techniques revolutionized the field of speaker recognition [25, 24, 12, 9]. Dramatic performance improvements were observed in both speed and accuracy. Over the past few years, error rates have decreased by a factor of five or more. At the same time, these new techniques have resulted in massive speed-ups, which have increased the scale of viable speaker-id systems by several orders of magnitude. These improvements stem from a recent shift in the speaker modeling paradigm. Only a few years ago, the model for each individual speaker was trained using data from only that particular speaker. Now, we make use of large speaker-labeled databases to learn distributions describing inter- and intra-speaker variability. This allows us to reveal the speech characteristics that are important for discriminating between speakers.

So-called i-vectors [9], where speech utterances are encoded into low dimensional fixed-length vectors that preserve information about speaker identity, further revolutionized the fields of speaker recognition. The concept of i-vectors, which now forms the basis of state-of-the-art systems, enabled new machine learning approaches to be applied to the speaker identification problem [9, 26, 7]. Inter- and intra-speaker variability can now be easily modeled using Bayesian approaches, which leads to superior performance [46, 5]. New training strategies can now benefit from the simpler statistical model form and the inherent speed-up [9, 19, 20].

The concept of subspace modeling, which form the basis of the aforementioned advances in speaker recognition, is also the focus of this habilitation thesis. The thesis, which takes form of commented collection of research paper, can be seen as a summary of my contribution to the topic. The included papers map my original work and the work done under my supervision or with my significant contribution.

## 1.1 Organization of the thesis

The thesis starts with a short introduction into speech processing and speaker verification, where we also discus the main challenge in this task – channel variability. The thesis continues with a short tutorial on subspace modeling and channel compensation techniques, which should make the collected research papers more accessible for a non-expert reader. The final part of the thesis is the collection of publications organized into three chapters:

Chapter 4 covers publications on the channel compensation techniques based on subspace modeling. The publications in this chapter describe and analyze different variants of Joint Factor Analysis model (see section 3.2) and their applications to not only speaker recognition (sections 4.2, 4.3, 4.7), but also to other speech processing problems such as language or emotion recognition [23, 29] (sections 4.5, 4.6). It is worth to note that a great deal of the success of the channel compensation techniques, which revolutionized speaker recognition and the related fields, has to be attributed to BUT Speech@FIT research group. In particular, I have developed the channel compensation techniques for STBU[1] systems [4] participating in the prestigious NIST Speaker Recognition Evaluations (SRE) [40]. In NIST SRE 2006 [37] (and the following evaluations), we have demonstrated the superior performance of the systems based on the channel compensation techniques described in the included papers (sections 4.2 and 4.3), which resulted in a broad acceptance and further development of these techniques by the scientific community. After the initial success with the new techniques, I have put together and led a group of top researchers from the speaker recognition field at Johns Hopkins University (JHU) Summer Workshop [8]. This research group made further major progress in the development of the subspace modeling techniques: The simplified fast scoring techniques, which made the new models even more appealing for practical applications were developed and described under my supervision in the PhD thesis by Ondřej Glembek [17]. A shorter description of the scoring techniques [18] is also included as the paper in section 4.4.

At the JHU Summer Workshop, i-vectors were also introduced [12], which are the focus of the publications collected in chapter 5. The included publications [34, 36, 10, 44] demonstrate the applicability of i-vectors to other than speaker recognition problems (sections 5.1, 5.2, 5.3, 5.7). Conceptually new approaches to fusion (section 5.5) [28] and to dicriminative training (section 5.4) [7] of speaker verification systems are described, which build on the concept of i-vectors. Originally, i-vectors were proposed to represent sequences of continuous feature vectors. The publications extending this concept to sequences of discrete features are also included [44, 27]. For this purpose, a new subspace multinomial model (section 5.6) and subspace n-Gram model (section 5.7) were proposed. A publication that introduces the nowadays popular technique for i-vector based discriminative adaptation of speech recognition system [10] is also included (section 5.3).

Chapter 6 deals with different extensions and modifications of the model for i-vector extraction. A simplified i-vector extraction model is proposed (section 6.1) in order to facilitate implementations of i-vector extraction into resource-limited embedded devices. Discriminative training of such simplified model is proposed (section 6.2) to compensate for the performance loss introduce by the approximations used. Finally,

[1] Consortium formed by Spescom DataVoice (South Africa), TNO (The Netherlands), Brno University of Technology (Czech Republic) and University of Stellenbosch (South Africa)

extensions of i-vector extractor robust to additive background noise are proposed [32, 35] in papers from sections 6.3 and 6.4.

# Chapter 2

# Basics of speech processing

## 2.1 Feature extraction

In speech processing, speech signal is typically represented by a sequence of speech frames: $\boldsymbol{O} = [\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T]$, where each frame $\boldsymbol{o}_t$ is a feature vector describing a short (typically 10 ms) stationary part of the signal. The standard speech features describing short-term spectral property of a frame are Mel Frequency Cepstral Coeefficients (MFCC) [11] or Predictive Linear Prediction (PLP) [22] coefficients.

## 2.2 Speaker verification

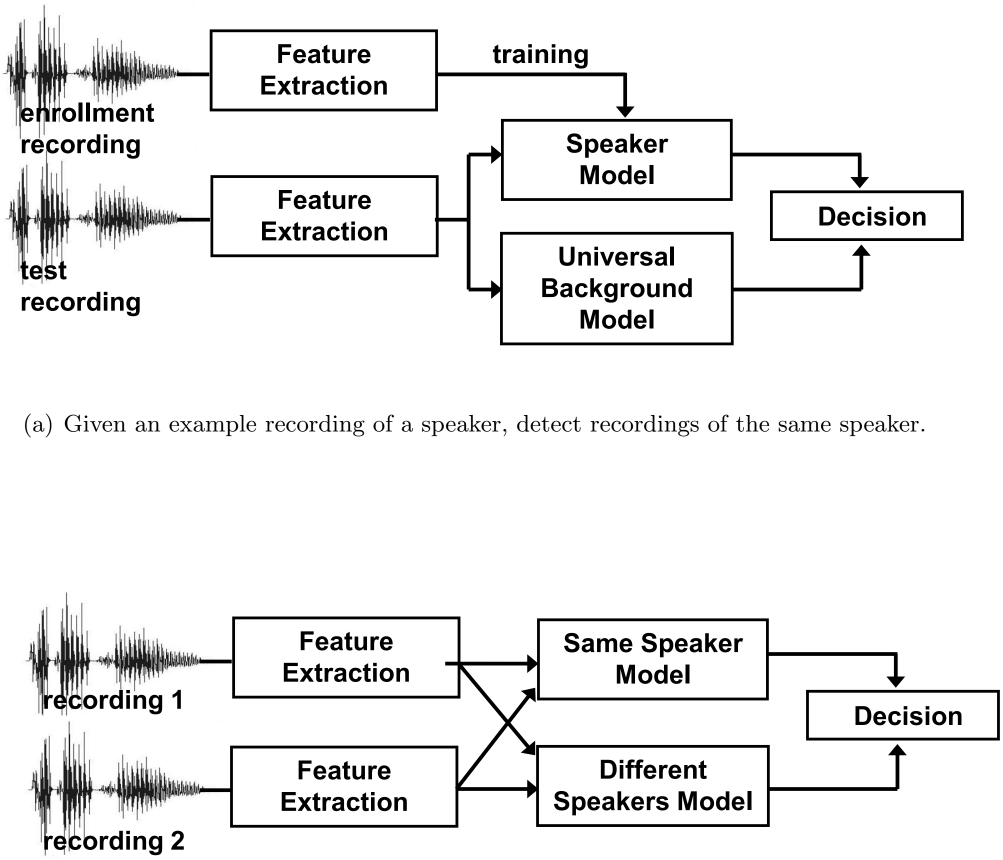
Many of the techniques described in this document were originally proposed for the task of speaker verification. In this section, the problem of speaker verification will be briefly described and the main challenges in this task will be discussed. We will also outline the basic scheme of speaker verification, which will serve as a starting point for development of the more advanced techniques described in this document.

Given an example recording(s) of a speaker, the task in speaker verification is to detect other recordings of the same speaker. Alternatively, the problem can be formulated as making a decision whether a pair of recordings (or more generally two sets of recording) comes from the same speaker or not. Although these are just two formulations of exactly the same problem, they roughly correspond to the two approaches depicted in Figure 2.1.

### 2.2.1 Traditional approach to speaker verification

The approach from Figure 2.1(a) is the traditional one, where a Universal Background Model (UBM) is trained on training data from many different speakers to model the general distribution of speech features. UBM is typically a Gaussian Mixture Model

$$p^{(UBM)}(\boldsymbol{o}_t) = \sum_{c=1}^{C} w_c \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{m}_c, \boldsymbol{\Sigma}_c), \tag{2.1}$$



(a) Given an example recording of a speaker, detect recordings of the same speaker.

(b) Given a pair of recordings, say whether these come from the same speaker.

Figure 2.1: Approaches to speaker verification.

where $C$ is a number of Gaussian components in the mixture (typically few thousands), $w_c$ are mixture weights and $\boldsymbol{m}_c$ and $\boldsymbol{\Sigma}_c$ are means and covariance matrices of the individual Gaussian components.

A speaker model describing the speaker-specific distribution $p^{(s)}(\boldsymbol{o}_t)$ is usually derived from the UBM by its adaptation to a speaker enrollment recording. A simple relevance MAP adaptation [14] was traditionally used for this purpose. In section 3.1, however, we will describe more recent Eigenvoice adaptation technique – an instance of the subspace modeling, which is the focus of this document.

To verify whether the test recording $\boldsymbol{O}_{(test)}$ comes from a target speaker $s$, speaker verification score can be calculated as a log likelihood ratio

$$score(\boldsymbol{O}_{test}, s) = \log \frac{p^{(s)}(\boldsymbol{O}_{test})}{p^{(UBM)}(\boldsymbol{O}_{test})}, \tag{2.2}$$

where the likelihoods of the feature sequences are calculated using the frame independency assumption simply as $p(\boldsymbol{O}) = \prod_t p(\boldsymbol{o}_t)$. To make the final binary speaker verification decision, the verification score $score(\boldsymbol{O})$ is compared to an appropriately set *decision threshold*.

## 2.3 Channel variability

An important challenge in speaker recognition is to deal with the intersession variability. Intersession variability is any variability in the speech signal that makes recordings of the same speaker to sound different. We can distinguish between *extrinsic* and *intrinsic* intersession variability. The *extrinsic* intersession variability can be attributed to the causes external to the speaker: different transmission channel (landline, cellular, VoIP, . . . ), microphones (electret, carbon-button, . . . ), acoustic environment and background noise (car, office, airport, restaurant, street, . . . ), and so on. The *intrinsic* intersession variability corresponds to the differences in speaker's voice caused, for example, by a variation in the vocal effort or by the speaker's emotional state (calm, nervous, stress, drunk, ill, . . . ). Since the variability attributable to the transmission channel is often considered to be the most relevant, we often talk only about *channel variability* or *channel* of a specific utterance. This term will, however, represent any of the variability causes mentioned above.

The channel variability can often cause variations in a speech signal energy that are larger than the differences caused by changing speakers. Therefore, it is important to introduce models that can model and decompose the variability in a signal into the useful between-speaker variability and the harmful channel variability. We will introduce such models in section 3.2.

To demonstrate the problem with channel variability, Figure 2.2 compares performances of the same speaker verification system on two conditions from NIST Speaker Recognition Evaluations (SRE) 2008 [38]. For both conditions, verification trials (the pairs of speech segments to be compared) consist of speech segments recorded over

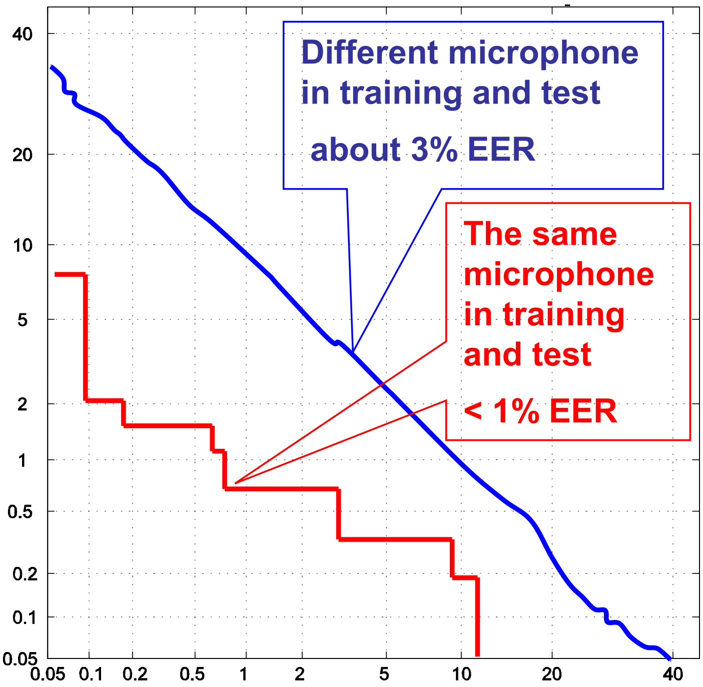


Figure 2.2: Degradation in speaker verification performance caused by microphone mismatch.

several different microphones placed in the recording room. For one of the conditions, however, the two recording in a verification trial always come from the same microphone, while for the other condition, a microphone mismatch is allowed. In Figure 2.2, the performance on the two conditions is compared in terms of Detection Error Tradeoff (DET) curve, showing tradeoff between probability of false alarms (false acceptance) and miss probability (false rejection) as obtained for different settings of the *decision threshold*. As can be seen, more than three times higher error rates should be expected when dealing with the microphone mismatch. It is important to note that the system in this experiment already employs the channel compensation techniques discussed in section 3.2. Without using any channel compensation techniques, the performance gap between the two conditions would be much larger as will be obvious from the results reported later in this document.

# Chapter 3

# Subspace modeling and channel compensation

A subspace model, as we understand it in the context of this document, is a generative statistical model, where the parameters of the model are constrained to live in a low dimensional subspace. A particular instance of the model (i.e. probability distribution represented by the model) can be then represented by a low-dimensional vector of coordinates in the subspace. In this chapter, an overview of the subspace modeling techniques and channel compensation techniques based on subspace modeling will be given in the chronological order of their development.

## 3.1 Eigenvoice adaptation

An early example of using such subspace model in speech processing is a technique called Eigenvoice speaker adaptation [30]. This technique was originally proposed for adapting the Hidden Markov Model (HMM) based speech recognition system to a particular speaker or to a specific acoustic environment. For simplicity, and to keep the continuity of the presentation, we will describe the Eigenvoice adaptation in the context of speaker verification as a technique for adapting UBM to a particular speaker [45] (i.e. enrolling a speaker model).

Using Eigenvoice adaptation, GMM specific to speaker $s$ can obtained by modifying the equation (2.1) as follows:

$$p^{(s)}(\boldsymbol{o}_t) = \sum_{c}^{C} w_c \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{m}_c + \boldsymbol{V}_c \boldsymbol{y}^{(s)}, \boldsymbol{\Sigma}_c), \tag{3.1}$$

where $w_c$, $\boldsymbol{m}_c$ and $\boldsymbol{\Sigma}_c$ are the UBM parameters, $\boldsymbol{V}_c$ are component-specific matrices describing subspace with large speaker variability and $\boldsymbol{y}^{(s)}$ is a speaker specific vector (or speaker factors). When setting the vector $\boldsymbol{y} = \boldsymbol{0}$, we recover the original UBM. In order to obtain good speaker model using an enrollment recording, we search for a speaker specific vector $\boldsymbol{y}^{(s)}$ that shifts the speaker specific means $\boldsymbol{\mu}_c^{(s)} = \boldsymbol{m}_c + \boldsymbol{V}_c \boldsymbol{y}^{(s)}$ to better match the distribution of the enrollment feature sequence. Note that there is no attempt to adapt other GMM parameters (i.e. the weights and covariance matrices), which will be also the case for the similar subspace models described later.

It will be instructive to define $\boldsymbol{\mu}^{(s)}$ as a vector that is constructed by concatenating all mean vectors $\boldsymbol{\mu}_c^{(s)}$ from all Gaussian components into one long speaker-specific mean super-vector. Similarly, we can construct a super-vector of UBM means $\boldsymbol{m}$ and a matrix $\boldsymbol{V}$ by stacking vectors $\boldsymbol{m}_c$ and matrices $\boldsymbol{V}_c$, respectively. The dimensionality of the resulting matrix $\boldsymbol{V}$ will be $FC \times R$, where $F$ is the dimensionality of speech features $\boldsymbol{o}_t$, $C$ is the number of Gaussian components and $R$ is the dimensionality of the speaker specific vector $\boldsymbol{y}^{(s)}$. Typically $R \ll FC$, and therefore $\boldsymbol{V}$ is a tall low-rank matrix. We can redefine Eigenvoice adaptation in terms of the super-vectors simply as

$$\boldsymbol{\mu}^{(s)} = \boldsymbol{m} + \boldsymbol{V}\boldsymbol{y}^{(s)}. \tag{3.2}$$

Now, it is easy to see that all the parameters in the high-dimensional vector $\boldsymbol{\mu}^{(s)}$ (typically hundreds of thousands of parameters) can be adapted to model a speaker specific distribution by estimating only small number of coefficients in the low-dimensional vector $\boldsymbol{y}^{(s)}$ (typically few hundreds of coefficients). This makes the Eigenvoice adaptation effective for very small amounts of adaptation data (few seconds) as opposed to other popular adaptation techniques (MLLR [31], MAP adaptation [33]), which need more data for adaptation to be effective.

### 3.1.1 Estimating Eigenvoices

In order to enroll a speaker model, the matrix $\boldsymbol{V}$ has to be estimated first. We would like to obtain $\boldsymbol{V}$ spanning a subspace of the mean super-vector space with a large between speaker variability. In other words, the columns of $\boldsymbol{V}$, which are called eigenvoices, should be bases capturing the correlations between the coefficients in mean super-vectors and they should point in the directions where speaker specific super-vectors vary the most. Probably the most straightforward way of estimating $\boldsymbol{V}$ [30] is to obtain *speaker specific models* for all speakers from the training data. Each *speaker specific model* can be obtained by simply re-training UBM on training data from one speaker, provided that reasonable amount of training data is available for each speaker. Preferably, *speaker specific model* is obtained by adapting UBM by means of another adaptation technique (typically MAP adaptation [33]). In the next step, mean super-vectors are extracted from each *speaker specific model*. Note, that it is important to keep the corresponding order of the mean components in the super-vectors extracted for different speakers. Finally, Principal Component Analysis (PCA) is applied to such super-vector dataset in order to find the subspace with the largest super-vectors variability (i.e. columns of $\boldsymbol{V}$ are given by eigenvectors corresponding to the largest eigenvalues of the covariance matrix estimated on the mean super-vectors).[1]

[1] Before applying PCA, it is also useful to normalize the individual means in the super-vectors by multiplying them with inverse square root of the corresponding covariance matrix. Inverse operation is then performed to “un-normalize” the resulting eigenvoices – collumns of $\boldsymbol{V}$. See [6] for details.

Eigenvoice adaptation was inspired by a similar technique used in face recognition, which was named Eigenfaces [43] – hence the name Eigenvoices. However, it is worth noting that for Eigenfaces, PCA is applied in the feature domain rather than in the domain of model parameters. Therefore, it is not an instance of subspace modeling as understood in this document.

### 3.1.2 ML estimation of speaker factors and eigenvoices

Once the matrix $\boldsymbol{V}$ is derived from training data, we can obtain the speaker adapted mean super-vector (and thus the speaker specific model) from equation (3.2) by properly estimating the speaker specific vector $\boldsymbol{y}^{(s)}$. As proposed in [30], $\boldsymbol{y}^{(s)}$ can be estimated to maximize the likelihood of the adaptation (enrollment) data:

$$\underset{\boldsymbol{y}^{(s)}}{\operatorname{argmax}}\, p^{(s)}(\boldsymbol{O}_{enroll}) = \underset{\boldsymbol{y}^{(s)}}{\operatorname{argmax}} \prod_{t=1}^{T_{enroll}} p^{(s)}(\boldsymbol{o}_t), \tag{3.3}$$

where $p^{(s)}(\boldsymbol{o}_t)$ is defined as in (3.1).

As an alternative to the PCA based estimation of $\boldsymbol{V}$, this matrix can be also estimated under the maximum likelihood (ML) framework. In [15], an iterative Expectation Maximization (EM) [13] based procedure is described, where vectors $\boldsymbol{y}^{(s)}$ are ML estimated, one for each speaker in the training data, using a fixed matrix $\boldsymbol{V}$. Then $\boldsymbol{V}$ is re-estimated to maximize the likelihood of the training data given the fixed speaker vectors. This procedure is iterated until convergence. Similar training procedure is also used in case of the channel compensation techniques described in the next section.

### 3.1.3 Eigenvoice adaptation for speech recognition

Eigenvoice adaptation was originally proposed to adapt HMM based speech recognition system to a particular speaker or a specific acoustic environment in order to improve its recognition performance. This technique can be applied to HMMs where probability distributions corresponding to HMM states are modeled by GMMs. It is straightforward to apply the model described above to such HMM base recognizer by simply forming the speaker specific mean super-vector using means from all Gaussian components from all HMM states. The detailed description of HMM as a generative model and the HMM based speech recognition is beyond the scope of this document, but the interested reader is kindly referred to [2, 48, 42].

It is worth to note, that Eigenvoices model the speaker variability, which is the harmful variability in the task of speech recognition. This is in contrast with speaker verification, where speaker variability is the useful variability. Therefore, Eigenvoices in speech recognition are in spirit very similar to Eigenchannels, which will be now introduced to cope with the unwanted variability in the speaker verification task.

## 3.2 Channel compensation

### 3.2.1 Simplfied JFA model

In [25, 24], Joint Factor Analysis (JFA) was proposed as a new model for speaker verification. This model can be seen as an extension of the Eigenvoice adaptation model, where a matrix of eigenchannels is introduced to model the unwanted channel variability. In fact, JFA is more complex model combining ideas of Eigenvoice adaptation and MAP adaptation [33] by treating speaker factors (and newly introduced channel factors) as probabilistic latent variables. As we will see, the model also allows for more theoretically sound definition of verification scores based on a more advanced Bayesian inference. However, to keep the presentation focused and comprehensible, we first introduce a simplified variant of the JFA model and we will later sketch its extensions towards its full version.

The Eigenvoice model (3.2) can be extended to represent speaker and channel specific distribution as

$$\boldsymbol{\mu}^{(sh)} = \boldsymbol{m} + \boldsymbol{V}\boldsymbol{y}^{(s)} + \boldsymbol{U}\boldsymbol{x}^{(h)}, \tag{3.4}$$

where $\boldsymbol{\mu}^{(sh)}$ is now speaker and channel specific mean super-vector, $\boldsymbol{U}$ is matrix of *eigenchannels* representing a subspace with a large channel (or intersession) variability in the mean super-vector space and $\boldsymbol{x}^{(h)}$ is a *channel specific* low-dimensional vector of channel factors. By *channel specific*, we usually understand *recording specific* as we usually assume that each recording comes from one channel and channel can change from recording to recording. Let us assume that we are given matrices of *eigenvoices* and *eigenchannels*, $\boldsymbol{V}$ and $\boldsymbol{U}$, that already well represent the subspaces with the large speaker and channel variability, respectively. By fixing $\boldsymbol{x}^{(h)}$ and trying different values of $\boldsymbol{y}^{(s)}$, we get different mean super-vectors corresponding to models of different speakers recorded over the same channel represented by the vector $\boldsymbol{x}^{(h)}$. Similarly, by fixing $\boldsymbol{y}^{(s)}$ and varying over different values of $\boldsymbol{x}^{(h)}$, we get models of the same speaker recorded over different channels.

### 3.2.2 Verification with JFA model

To use this model for speaker verification, we can proceed as follows: Both speaker factors $\boldsymbol{y}^{(s)}$ and channel factors $\boldsymbol{x}^{(h)}$ are estimated to maximize likelihood of an enrollment recording. This way, we obtain speaker model, which is not only specific to the speaker but also to the channel of the enrollment recording. To score the speaker model against a test recording, channel factors $\boldsymbol{x}^{(h)}$ are first ML re-estimated on the test recording, which corresponds to adapting the speaker model to the channel of the test recording. Finally, we can evaluate likelihood of the test recording $p^{(s)}(\boldsymbol{O}_{test})$ from equation (2.2) using the speaker and channel adapted model. To evaluate the complete log likelihood ratio verification score (2.2), we also need the likelihood from the denominator $p^{(UBM)}(\boldsymbol{O}_{test})$. This is usually evaluated using a model where $\boldsymbol{y}^{(s)} = 0$ and $\boldsymbol{x}^{(h)}$ is adapted to the test recording (i.e. UBM adapted to the test recording channel).

### 3.2.3 JFA model estimation

Before using the proposed model for verification, we need to estimate matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ on training data. It is also possible to re-estimate the remaining model parameters (i.e. $\boldsymbol{m}$ and GMM component weights and covariance matrices) together with $\boldsymbol{V}$ and $\boldsymbol{U}$. However, it was found that these parameters can be simply taken from the pre-trained UBM without sacrificing any verification performance. To estimate the matrices $\boldsymbol{V}$ and $\boldsymbol{U}$, we use a procedure similar to the one described in section 3.1.2, where we iterate between ML estimation of $\boldsymbol{y}^{(s)}$ and $\boldsymbol{x}^{(h)}$ for fixed $\boldsymbol{V}$ and $\boldsymbol{U}$, and the other way around, starting from a randomly initialized $\boldsymbol{V}$ and $\boldsymbol{U}$. The training data should comprise recordings of many speakers each recorded in several sessions. During training, there is one vector $\boldsymbol{y}^{(s)}$ for each training speaker $s$ and one $\boldsymbol{x}^{(h)}$ for each training recording. In other words, speaker factors $\boldsymbol{y}^{(s)}$ are constrained to be the same for all recordings of the same speaker, while different channel factors $\boldsymbol{x}^{(h)}$ are estimated for each individual training recording.

### 3.2.4 Full JFA model

So far, we have described JFA as a model where the speaker and channel factors, $\boldsymbol{y}^{(s)}$ and $\boldsymbol{x}^{(h)}$, are parameters of the model, which can be estimated under the maximum likelihood framework. In the original JFA model [25, 24], however, speaker and channel factors are treated as latent random variables having standard normal prior distributions

$$\begin{aligned} p(\boldsymbol{y}) &= \mathcal{N}(\boldsymbol{y}; \boldsymbol{0}, \boldsymbol{I}) && (3.5) \\ p(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I}). && (3.6) \end{aligned}$$

We rewrite the JFA equation for mean super-vector using the latent variables:

$$\boldsymbol{m} = \boldsymbol{\mu} + \boldsymbol{V}\boldsymbol{y} + \boldsymbol{U}\boldsymbol{x}. \tag{3.7}$$

This equations defines $\boldsymbol{m}$ as a random variable in terms of the random variables $\boldsymbol{y}$ and $\boldsymbol{x}$. Therefore, JFA model can be seen as a two-level generative model assuming that a sequence of speech features is generated from a GMM whose mean super-vector is first itself generated from (3.7). We can interpret this equation as a model making LDA-like assumptions about mean super-vectors. In particular, the across-speaker (across-class) distribution of super-vectors is assumed to be $\boldsymbol{\mu} + \boldsymbol{V}\boldsymbol{y}$, which corresponds to a Gaussian distribution with a global mean $\boldsymbol{\mu}$ and across-speaker covariance matrix $\boldsymbol{V}\boldsymbol{V}^T$. The within-speaker (within-class) covariance matrix is then given by $\boldsymbol{U}\boldsymbol{U}^T$. Note that, since $\boldsymbol{V}$ and $\boldsymbol{U}$ are typically low-rank matrices, the covariance matrices $\boldsymbol{V}\boldsymbol{V}^T$ and $\boldsymbol{U}\boldsymbol{U}^T$ will be also low-rank. Therefore, the mean super-vectors will be Gaussian distributed only in the subspace spanned by basis $\boldsymbol{V}$ and $\boldsymbol{U}$.

The main advantage of this new probabilistic definition of JFA model is that the bases $\boldsymbol{V}$ and $\boldsymbol{U}$ not only represent the sub-spaces in which mean super-vectors live, but they also represent the amounts of within-speaker and across-speaker variability in these subspaces (i.e. they represent the corresponding covariance matrices). Specifically, each column of $\boldsymbol{V}$ is a vector pointing to a direction with a large across-speaker variability in the super-vectors space and the magnitude (length) of this vector represents the standard deviation in this direction. The matrix $\boldsymbol{U}$ can be interpreted similarly in terms of within-speaker variability.

In the original JFA model [25, 24]

$$\boldsymbol{m} = \boldsymbol{\mu} + \boldsymbol{V}\boldsymbol{y} + \boldsymbol{U}\boldsymbol{x} + \boldsymbol{\epsilon}, \tag{3.8}$$

one more factor $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{D}^2)$ can be found, which is a random variable describing the residual speaker variability not covered by $\boldsymbol{V}$.[2] With this term, we obtain full-rank across-speaker covariance $\boldsymbol{V}\boldsymbol{V}^T + \boldsymbol{D}^2$, which corresponds to the standard factor analysis model [2].[3] This allows super-vectors to live outside of the subspaces defined by $\boldsymbol{V}$ and $\boldsymbol{U}$. However, the residual variability represented by $\boldsymbol{\epsilon}$ is typically very small, which means that it is very unlikely for a super-vector to be far from those subspaces. In real applications, however, inclusion of $\boldsymbol{\epsilon}$ does not seem to have any practical advantage, at least when dealing with text independent speaker verification and recordings containing no more than few minutes of speech. Therefore, we omit this term in the following discussion.

The parameters of the full JFA model can be ML estimated using a similar iterative EM algorithm as described in the previous section. However, rather than taking the point estimates of the latent variables, the training algorithm can consider their full posterior distributions. A detailed description of the training procedure is out of the scope of this brief introduction and the kind reader is referred to [25, 3] for more details.

### 3.2.5 Inference in the full JFA model

In most of the practical speaker verification systems based on the full JFA model, the model is used to infer point estimates of speaker and channel factors $\boldsymbol{y}^{(s)}$ and $\boldsymbol{x}^{(h)}$ in very much the same way as described in section 3.2.2. However, given the priors on the latent variables, $\boldsymbol{y}$ and $\boldsymbol{x}$ and given a recording $\boldsymbol{O}$, we can infer posterior distribution of the factors

$$p(\boldsymbol{y}, \boldsymbol{x}|\boldsymbol{O}) \propto p(\boldsymbol{O}|\boldsymbol{y}, \boldsymbol{x})p(\boldsymbol{y})p(\boldsymbol{x}), \tag{3.9}$$

where $p(\boldsymbol{O}|\boldsymbol{y}, \boldsymbol{x})$ is a likelihood calculated for a GMM corresponding to a specific setting of factors $\boldsymbol{y}$ and $\boldsymbol{x}$. As the point estimates for $\boldsymbol{y}^{(s)}$ and $\boldsymbol{x}^{(h)}$, we can now select their most probable values based on the posterior (3.9). In other words, we use maximum a-posteriori (MAP) estimates of $\boldsymbol{y}^{(s)}$ and $\boldsymbol{x}^{(h)}$ rather than ML estimates proposed in section 3.2.2. Equivalently, we can say that we obtain the most probable super-vectors (and thus GMM) specific to the the speaker and channel of the recording $\boldsymbol{O}$. Note that

[2] $\boldsymbol{\epsilon}$ is often represented as $\boldsymbol{D}\boldsymbol{z}$, where $\boldsymbol{D}$ is diagonal square matrix and $\boldsymbol{z}$ is standard normal distributed latent vector – so called common factors

[3] Although, here, factor analysis is applied in the space of GMM model parameters rather than directly to the observed data, which is more common.

unlike ML estimation, MAP estimation takes into account the information about the amount of variability in different directions of the super-vector space as encoded in the matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ and as learned from the training data. Loosely speaking, a mean super-vector can easily move in the high-variance directions, while a lot of support from data is necessary to move the super-vector in a direction with low speaker or channel variance. This way, JFA model combines the ideas of Eigenvoice adaptation and MAP adaptation [33].

As before, to evaluate a speaker model on a test recording, we can fix $\boldsymbol{y}^{(s)}$ and adapt the model to the channel of the test recording by obtaining new MAP point estimate of $\boldsymbol{x}^{(h)}$. With the latent variables, however, the likelihood from numerator of (2.2) can be evaluated in more principled and theoretically sound way as

$$p^{(s)}(\boldsymbol{O}_{test}) = p(\boldsymbol{O}_{test}|\boldsymbol{y}^{(s)}) = \int p(\boldsymbol{O}_{test}|\boldsymbol{y}^{(s)}, \boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \tag{3.10}$$

where we take into account any possible channel of the test recording by integrating over the channel factors. Similarly, to enroll speaker factors while taking into account any possible channel of the enrollment recording $\boldsymbol{O}_{enroll}$, we should find $\boldsymbol{y}^{(s)}$ maximizing posterior distribution

$$p(\boldsymbol{y}|\boldsymbol{O}_{enroll}) = \int p(\boldsymbol{y}, \boldsymbol{x}|\boldsymbol{O}_{enroll})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \tag{3.11}$$

where we integrated out the channel factors from equation (3.9).

Finally, the log likelihood score that is best theoretically justified for this model can be calculated as

$$score(\boldsymbol{O}_{test}, \boldsymbol{O}_{enroll}) = \log \frac{\int p(\boldsymbol{O}_{test}|\boldsymbol{y})p(\boldsymbol{y}|\boldsymbol{O}_{enroll})\mathrm{d}\boldsymbol{y}}{\int p(\boldsymbol{O}_{test}|\boldsymbol{y})p(\boldsymbol{y})\mathrm{d}\boldsymbol{y}}, \tag{3.12}$$

where, in the numerator, we evaluate likelihood of the test recording for an enrolled speaker like in (3.10). However, now we integrate over any possible speaker model as represented by the posterior distribution of speaker factors $p(\boldsymbol{y}|\boldsymbol{O}_{enroll})$. The denominator corresponds to the likelihood of the test recording “given any speaker” (i.e. we integrate over the prior distribution of $\boldsymbol{y}$). It can be shown, that the score (3.12) can be equivalently expressed in terms of so called Bayes factors as

$$score(\boldsymbol{O}_{test}, \boldsymbol{O}_{enroll}) = \log \frac{\int p(\boldsymbol{O}_{test}|\boldsymbol{y})p(\boldsymbol{O}_{enroll}|\boldsymbol{y})p(\boldsymbol{y})\mathrm{d}\boldsymbol{y}}{p(\boldsymbol{O}_{test})p(\boldsymbol{O}_{enroll})}, \tag{3.13}$$

where the terms in the denominator can be evaluated as $p(\boldsymbol{O}) = \int p(\boldsymbol{O}|\boldsymbol{y})p(\boldsymbol{y})\mathrm{d}\boldsymbol{y}$. The numerator in (3.13) is the likelihood of the hypothesis that both the enrollment and the test recording were produced by the same speaker, while the denominator is the likelihood of generating $p(\boldsymbol{O}_{test})$ and $p(\boldsymbol{O}_{enroll})$ independently (i.e from two different speakers). Such scoring now corresponds to the scheme from figure 2.1(b), where likelihoods from two models representing same-speaker and different-speaker hypotheses are compared. Note the symmetrical role of both recordings in (3.13) (i.e.

the score does not change when switching roles of recordings $\boldsymbol{O}_{test}$ and $\boldsymbol{O}_{enroll}$). This is in contrast to (2.2), where speaker model is trained on one recording and evaluated on the other one.

Unfortunately, it is intractable to analytically evaluate the score (3.13). A Variational Bayes inference was proposed [2, 49] to approximate the score. However, this inference was not adopted in the practical applications as it is computationally expensive to evaluate and leads to very limited improvements over the more approximate inference described in the beginning of this section.

More details on JFA model scoring are given in the included paper 4.4, where some of the approximations described above are compared in terms of computational cost and verification performance. In this paper, it was found that very crude approximations can be implemented, allowing for extremely fast score evaluation without sacrificing any verification performance. Such scoring made JFA even more appealing for the practical application. It is worth to note that, regardless the exact way of calculating verification scores, rather ad-hoc normalization techniques such as t-norm or z-norm [1, 47, 9] are necessary to calibrate scores for different speakers and/or test utterances in order to obtain good verification performance. This is the case even when the score is calculated in the theoretically correct way according to equation (3.13).

## 3.3 i-vectors

Although JFA model provided an excellent verification performance compared to the earlier techniques, the effectiveness of all the approximations and the ad-hoc normalizations in the score evaluation makes the validity of JFA as a proper generative model questionable. Since the point MAP estimates of speaker factor $\boldsymbol{y}^{(s)}$ were found sufficient to represent a speaker model, they must contain enough of the relevant information about the speaker of the corresponding recording. This led us to the idea of performing verification based on a mere comparison of the speaker factors extracted from the two recordings in a verification trial. At JHU 2008 summer workshop [8], the experiments were carried out where speaker factors were used as a low-dimensional, fixed-length features representing individual recordings. It was soon discovered that not only the speaker factors but also the channel factors estimated on a recording contain a considerable amount of speaker specific information. This finally led to the proposal of i-vectors as a feature extraction technique, where each recording is represented by a low-dimensional, fixed-length vector.

The model for extracting i-vectors is essentially the same as JFA model, except that it comes only with one subspace $\boldsymbol{T}$ describing all the inter-recording variability comprising both the speaker and channel variability:

$$\boldsymbol{m} = \boldsymbol{\mu} + \boldsymbol{T}\boldsymbol{w}. \tag{3.14}$$

The subspace $\boldsymbol{T}$ is sometimes referred to as *total variability subspace*. The procedure for model training is also essentially the same as for JFA, albeit simpler, as there are no speaker factors, only the recording specific latent variables $\boldsymbol{w}$, which can be inferred independently one for each recording. Therefore, there is no need for having speaker labeled training data and the subspace $\boldsymbol{T}$ can be trained in an unsupervised way on a large amount of unlabeled recordings. It is only assumed that each training recording contains speech of only one speaker.

As was already said, the main conceptual difference between using i-vector and JFA models is that the i-vector model is used only as a feature extractor. It is not used to evaluate any verification scores as in the case of JFA. Instead, *i-vector* is obtained for each recording as a MAP point estimate of the latent variable $\boldsymbol{w}$. It is used as a low-dimensional feature vector (typically few hundreds of dimensions) representing the recording.

Since the i-vector model makes no attempt to separate the speaker and channel specific variability, the extracted i-vector contains information about both speaker and channel. This needs to be handled by the following back-end classification model, which is used to produce the final verification score. The concept of i-vectors opened the door for experimenting with different and often very simple back-end classifiers: In the original work on i-vectors [12], cosine distance metric together with the *within-class covariance normalization* (WCCN) [21] was found to produce quality verification scores outperforming JFA model. Currently, Probabilistic Linear Discriminant Analysis (PLDA) [41, 26, 7] is considered the state-of-the-art model for i-vector based speaker verification. This model makes LDA-like assumptions similar to those described for JFA in section 3.2.5. However, now the model is applied in the i-vector (feature) domain rather than in the domain of mean super-vectors. In this simpler case, the proper Bayesian inference for verification score evaluation (3.13) is analytically tractable and, in fact, computationally very efficient as described in the included paper 5.4.

## 3.4 Effectiveness of Channel compensation techniques

Performances of the different channel compensation techniques introduced in this section are compared in figure 3.1. The figure shows DET curves for systems evaluated on the data from NIST SRE 2010, condition 5 (telephone-telephone trials) [39]. As a baseline, we see the performance for the system based on *relevance MAP adaptation*, which was the state-of-the-art technique before the introduction of the subspace-based channel compensation techniques. This system already uses some earlier techniques to cope with the problem of channel mismatch, most of which became obsolete after the introduction of the subspace-based techniques as demonstrated in the included paper 4.2.

Eigenchannel adaptation is a simplified variant of JFA system, where speaker models are still enrolled using the *relevance MAP adaptation*, but afterward the models are adapted to the channel of a test utterance using the eigenchannel subspace as in the JFA model. Also, a PCA-based estimation of the eigenchannel subspace is used that is similar to the one described for Eigenvoice adaptation in section 3.1.1. The detailed description of this system is given in the included papers 4.2 and 4.3. These papers describe the system submitted into NIST SRE 2006 evaluations demonstrating the effectiveness of the subspace-based channel compensation, which resulted in the broad acceptance of these techniques by the scientific community.

As can be seen from figure 3.1, full version of the JFA model as described in section 3.2.5 outperforms the simpler Eigenchannel adaptation method. Different variants of JFA model are analyzed in more detail in the included paper 4.1. The concept of i-vectors brought additional significant improvements especially after introduction of i-vector postprocessing steps known as length normalization [16].

In this text, we have presented the channel compensation techniques and i-vectors applied to the task of speaker verification. Although originally proposed and developed for this task, they quickly found their way into different related fields of speech processing. Applications of these techniques to various problems (e.g. language recognition, emotion recognition, speaker adaptation for speech recognition, . . . ) is covered by the following included papers.

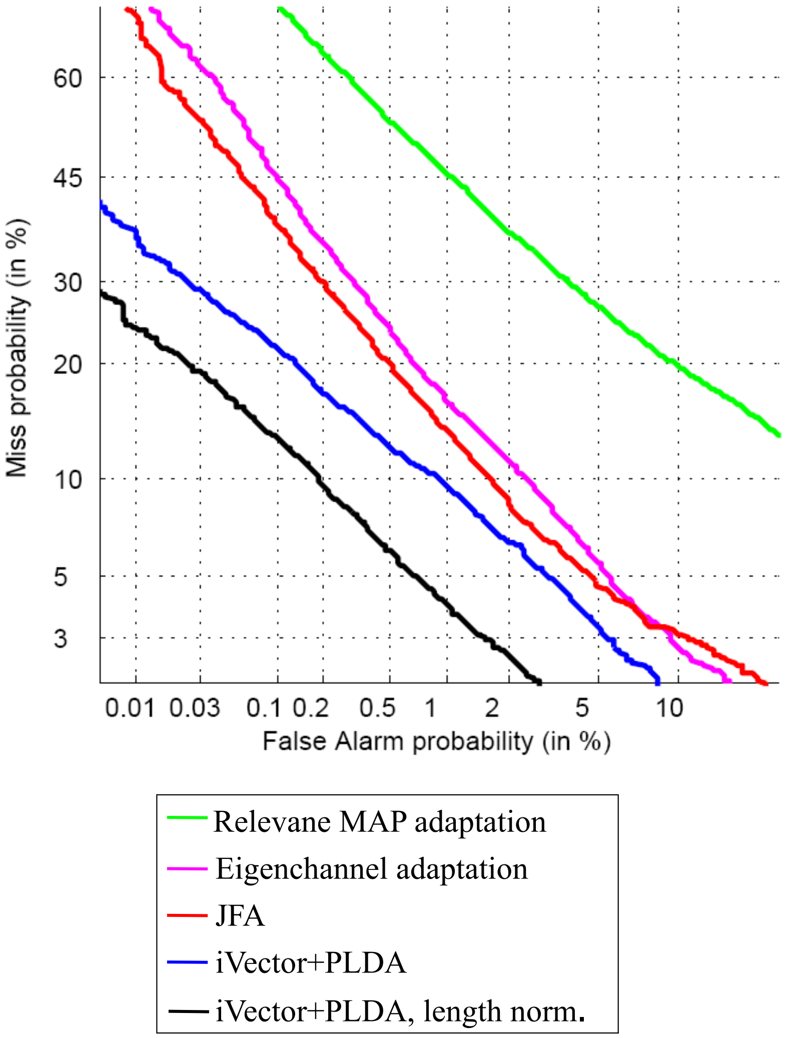


Figure 3.1: Comparison of different channel compensation techniques on NIST SRE 2010 condition, 5 (telephone-telephone trials) task.

# Chapter 4

# Applications of Channel Compensation

This chapter covers publications on channel compensation techniques based on subspace modeling. They describes and analyze different variants of Joint Factor Analysis model (sections 4.1 and  4.4) and their applications not only to speaker recognition (sections 4.2, 4.3, 4.7), but also to other speech processing problems such as language or emotion recognition (sections 4.5, 4.6). The included papers 4.2 and 4.3 describe the very successful system participating in NIST SRE 2006 evaluations, which introduced the channel compensation techniques to the broad scientific community.

INTERSPEECH 2009 BRIGHTON

# Investigation into variants of Joint Factor Analysis for speaker recognition

Lukáš Burget, Pavel Matějka, Valiantsina Hubeika and Jan “Honza” Černocký

Speech@FIT, Brno University of Technology, Czech Republic
{burget|matejkap|ihubeika|cernocky}@fit.vutbr.cz

## Abstract

In this paper, we have investigated into JFA used for speaker recognition. First, we performed systematic comparison of full JFA with its simplified variants and confirmed superior performance of the full JFA with both eigenchannels and eigenvoices. We investigated into sensitivity of JFA on the number of eigenvoices both for the full one and simplified variants. We studied the importance of normalization and found that gender-dependent zt-norm was crucial. The results are reported on NIST 2006 and 2008 SRE evaluation data.

**Index Terms**: speaker recognition, joint factor analysis.

## 1. Introduction

Nowadays speaker recognition systems are usually based on Universal Background Model-Gaussian Mixture Modeling (UBM-GMM) and employ a number of techniques that improve GMM modeling capability and help fight against the main problem in speaker verification - the inter-session variability. This is caused by differences in channels, acoustic conditions and other factors varying across the speech segments being compared [2]. In several past years, systems based on Joint Factor Analysis (JFA) [4] obtained wide attention due to their ability to explicitly model the inter-session variability. However, different research labs adopted different variants JFA and it was unclear how do these variants compare in terms of recognition performance. The aim of this paper is to provide the comparison of such JFA variants and give some insight into the process of building state-of-the-art JFA system.

JFA model is a two-level generative model assuming that speech segments are generated from a GMM whose mean super-vector $\mathbf{M}$ – vector of concatenated GMM means – is first itself generated from the following distributions:

$$\mathbf{M} = \mathbf{m} + \mathbf{V}\mathbf{y} + \mathbf{D}\mathbf{z} + \mathbf{U}\mathbf{x}, \qquad (1)$$

where $\mathbf{m}$ is speaker-independent mean super-vector, $\mathbf{U}$ is a subspace with high intersession variability (eigenchannels[1]), $\mathbf{V}$ is a subspace with high speaker variability (eigenvoices) and $\mathbf{D}$ is a diagonal matrix describing remaining speaker variability not covered by $\mathbf{V}$. Speaker factors $\mathbf{y}$, $\mathbf{z}$ and channel factors $\mathbf{x}$ are assumed to be normally distributed random variables. For segments of the same speaker, speaker factors are assumed to be the same, while channel factors are allowed to differ. For details, we recommend Kenny's paper [4] that served us as inspiration for building the baseline JFA systems presented in this paper.

The results in this paper are presented on NIST SRE 2006 evaluation data, especially the 1conv4w-1conv4w all-trials condition (det1 – tel-tel). The sets for other conditions (tel-mic, mic-tel, mic-mic) were defined by MIT-LL and are described in [7].

## 2. Baseline systems

As a baseline for the analysis presented in this paper, we have chosen two JFA systems developed for NIST SRE 2008 evaluations. The two systems differs mainly in the feature extraction.

The first system is based on features that are short time gaussianized MFCC 12 + C0 augmented with their delta, double delta and triple delta coefficients. The dimensionality of the resulting features is reduced from 52 to 39 using HLDA. HLDA classes correspond to UBM Gaussians. These features were previously used in our NIST SRE 2006 submission [2]. The system based on these features will be denoted **MFCC13⇒39**.

Inspired by the outstanding performance of the system described in [4], features used for our second baseline system are short time gaussianized MFCC 19 + energy augmented with their delta and double delta coefficients, making 60 dimensional feature vector without any dimensionality reduction . The system making use of these features will be denoted **MFCC20⇒60**.

In both cases, the features are derived with classical analysis window of 20 ms with shift of 10 ms and short-time gaussianization using window of 300 frames (3 sec). Speech/silence segmentation is performed by our Hungarian phone recognizer [1, 2], where all phoneme classes are linked to 'speech' class. Several heuristics based on short-term energy are used for two-channel telephone data to eliminate cross-talks [2].

The training of the JFA systems closely follows the description of “Large Factor Analysis model” in Patrick Kenny's paper [4]. First, UBM model with 2048 Gaussian components is trained using Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, and NIST SRE 2004 and 2005 telephone data, which is in turn used to collect zero and first order statistic for training the JFA systems. The mean super-vector $\mathbf{m}$ from (1) was set to the UBM mean and on contrary to [4] was never re-trained. The variances of Gaussian components are also taken from UBM and not re-trained in the training of JFA.

First, for each JFA system, 300 eigenvoices (matrix $\mathbf{V}$) are trained using EM algorithm [4] on the same data as UBM. For the estimated eigenvoices, MAP estimates of speaker factors are obtained and fixed for the following training of eigenchannels. A set of 100 eigenchannels is trained on NIST SRE 2004

This work was partly supported by European projects MOBIO (FP7-214324) and AMIDA (FP6-033812), by Grant Agency of Czech Republic project No. 102/08/0707, and by Czech Ministry of Education project No. MSM0021630528. We would like to thank MIT-LL team for creating the sets with the microphone conditions and Patrick Kenny for fruitful discussions helpful advices

[1] We refer to “eigenvoices” and “eigenchannels” following the terminology defined in [4] although these sub-spaces are estimated using EM-algorithm, not PCA.

and 2005 telephone data. Another set of 100 eigenchannels is trained on SRE 2005 auxiliary microphone data to allow the system to deal with the microphone speech segments. Both sets are stacked to form the final matrix $\mathbf{U}$. On contrary to Kenny's paper [4], the diagonal matrix describing the remaining speaker super-vector variability (matrix $\mathbf{D}$ in (1)) is estimated on top of eigenvoices and eigenchannels. A small disjoint set of NIST SRE 2004 speakers (recordings of only 44 females and 13 males) is used for training of $\mathbf{D}$ using fixed MAP point estimates of speaker and channel factors. To obtain speaker models, MAP point estimates of all the factors are estimated on enrollment segments using Gauss-Seidel-like iterative method [6]. For details about the training data and its splits for training the different sets of hyperparameters see [7]. In all the experiments described in this paper, the standard 10-best Expected Log Likelihood Ratio frame-by-frame scoring was used. It was based on the MAP point estimates of the channel factors[2].

Unless stated otherwise, all results were obtained with scores normalized using zt-norm. We have used 221 females and 149 males z-norm segments, 200 females and 159 males t-norm models, together 729 segments taken each from one speaker of NIST SRE 2004 and 2005 data.

In the case of systems developed for NIST SRE 2008 evaluations, single gender-independent (GI) system **MFCC13⇒39** was trained and evaluated using the data of both genders, while two gender-dependent (GD) systems **MFCC20⇒60** were trained and evaluated using the data of only the corresponding gender. However, note that gender dependent zt-norm was applied in both cases (i.e. even for system **MFCC13⇒39**, only z-norm segments and t-norm models of corresponding gender were used to normalize scores). The performance of these systems is demonstrated in Fig 1. On the left, we can see that the larger (GD, feature dimensionality 60) system **MFCC20⇒60** outperforms the smaller (GI, feature dimensionality 39) system **MFCC13⇒39** when evaluating on tel-tel condition. To see, whether the improvement comes from using GD models or from using different features, we have also trained GI version of **MFCC20⇒60** system, which is also shown in the figure. It seems that most of the improvement comes from the features with more detailed spectral resolution as the performances of both GD and GI versions are comparable. However, for low false-alarm region, which is the region of main interest in NIST evaluations, performance of the GD system is superior. Conversely, **MFCC13⇒39** system performs better on mic-mic trials shown on the right panel in Fig 1. The most probable reason for it is that large **MFCC20⇒60** system is overtrained to telephone data, which is the only type of data used for training UBM and speaker subspace hyperparameters. This hypothesis is also supported by the improved performance of **MFCC20⇒60** system when halving the number of system parameters by using GI instead of GD version. Unless stated otherwise, the GI version of **MFCC20⇒60** system will be used in the following experiments.

## 3. Analysis of JFA

### 3.1. Variants of Joined Factor Analysis

In the past years, different research labs adopted simplified variants of full JFA dropping some of the terms in (1) and using different methods for the hyperparameter estimation. In this section, we present a comparison of some of the JFA variants and we show that the baseline (full JFA) systems provide superior performance. Systems with only 50 eigenchannels are used in these experiments to allow for fair comparison as this was found to be the optimal number of eigenchannels for the simplified JFA variants described here.

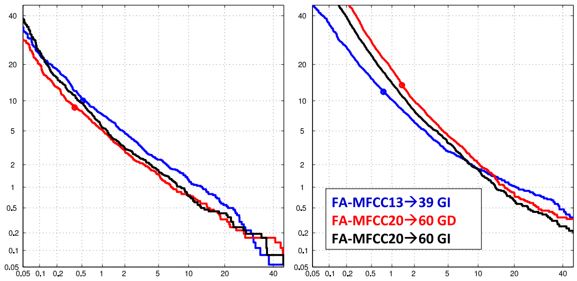


Figure 1: Performance of JFA systems based on different features and gender dependent or gender independent variants. Results on NIST 2006 data. Left panel: tel-tel trials, right panel: mic-mic trials.

#### 3.1.1. *Relevance MAP adaptation*

The standard relevance MAP adaptation [9] can be actually seen as a special simplest case of JFA. Dropping the terms with eigenvoices and eigenchannels in equation (1), we obtain $\mathbf{M} = \mathbf{m} + \mathbf{Dz}$. For relevance MAP we simply set $\mathbf{D}^2 = \mathbf{\Sigma}/\tau$, where $\mathbf{\Sigma}$ is diagonal matrix with super-vector of UBM variances in the diagonal and $\tau$ is the relevance factor. For point MAP estimates of factors $\mathbf{z}$, it is then easy to show that the speaker model represented by $\mathbf{M}$ is equivalent to that obtained with standard relevance MAP re-estimation formulae [9].

#### 3.1.2. *Eigenchannel adaptation*

The systems with *eigenchannel adaptation* [3, 2] use relevance MAP for enrolling speaker model. In the test phase, each speaker model is MAP adapted to the channel of test utterance by estimating the channel factors $\mathbf{x}$. Unlike the case of other JFA variants, PCA is used to estimate the eigenchannel matrix $\mathbf{U}$ instead of the EM algorithm. No eigenvoices are considered by this system. See [2] for thorough description of *eigenchannel adaptation* and its comparison with a system without channel compensation.

#### 3.1.3. *JFA without eigenvoices with relevance-MAP-like* $\mathbf{D}$

In [6, 8], JFA systems without eigenvoices are described, where only the eigenchannel matrix $\mathbf{U}$ is trained using EM algorithm on top of the $\mathbf{D}$ matrix, which is set as in the case of the relevance MAP. On contrary to the system system based on *eigenchannel adaptation*, here, the inter-session variability is considered also for enrollment. In both [6] and [8], given the enrollment segment, MAP point estimates of factors $\mathbf{z}$ and $\mathbf{x}$ are estimated jointly using Gauss-Seidel-like iterative method. The processing of a test segment is the same as for *eigenchannel adaptation*.

#### 3.1.4. *JFA without eigenvoices with* $\mathbf{D}$ *matrix trained on data*

As an alternative to the previous JFA variant, the $\mathbf{D}$ matrix in systems without eigenvoices can be also trained using EM algorithm (see the system with zero speaker factors in [4]). In

[2] Note that in [10], we have shown that similar or better results can be obtained with different approximate scoring schemes, while significantly speeding up the scoring process.

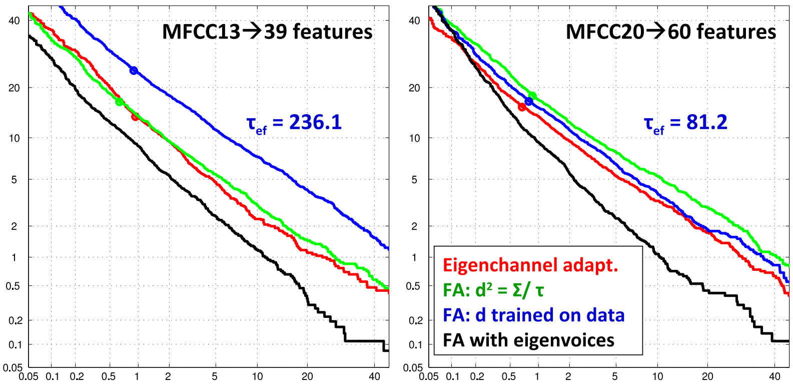


Figure 2: Flavors of JFA. Results on NIST 2006 data.

our experiment with this system, $\mathbf{D}$ matrix is trained first (unlike for the baseline system) and it is fixed for the following training of the eigenchannel matrix $\mathbf{U}$. Note also that all the data that are used for training eigenvoices in the baseline system, are now used for training $\mathbf{D}$. In the case of relevance MAP, the relevance factor $\tau$ has an intuitive interpretation. It specifies the number of frames in the adaptation data associated with a given UBM Gaussian component, which makes the MAP adaptation to shift the Gaussian component right in a half way between its original position and mean of the adaptation data. Training the matrix $\mathbf{D}$ from the data can be seen as training specific relevance factor for each coefficient of each Gaussian component. As proposed by Kenny, effective relevance factor $\tau_{ef} = trace(\mathbf{\Sigma})/trace(\mathbf{D}^2)$ can be used in this case, which can be loosely interpreted as a number of frames needed in average for each component to make the adaptation effective.

### *3.1.5. Results with JFA variants*

The results on NIST 2006 data obtained with the JFA variants described above are shown in Fig 2. All the JFA variants without eigenvoices provide comparable performance for both types of features MFCC13⇒39 features and MFCC20⇒60. The simple eigenchannel adaptation seems to be somewhat more robust, though. The exception is the system with $\mathbf{D}$ trained on features MFCC13⇒39, which fails to perform well. The effective relevance factor $\tau_{ef} = 236.1$ for this system is significantly higher than for MFCC20⇒60 ($\tau_{ef} = 81.2$), which probably prevented the system to effectively adapt to enrollment data. The reason for this failure is still unclear and deserves further investigation. Finally, the full JFA system with eigenvoices significantly outperforms all the other JFA configurations on both feature sets.

### 3.2. Sensitivity of JFA to the number of eigenchannels

In Fig. 3, the three solid lines show again the performance of three JFA variants from the previous section, where 50 eigenchannels were trained for each system. The dashed lines show the change in the performance with increased number of 100 eigenchannels. We observe degradation in performance for the two variants without eigenvoices, namely the eigenchannel adaptation and the JFA with $\mathbf{D}$ trained on data. These systems seem not to be able to robustly estimate the increased number of eigenchannels. However, in the case of full JFA system, we benefit from more eigenchannels significantly after explaining the speaker variability in the model space by eigenvoices.

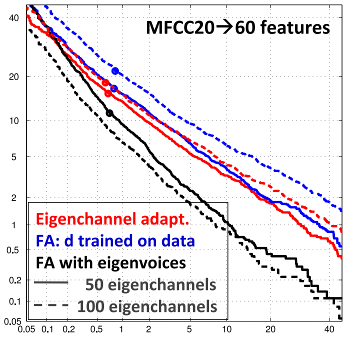


Figure 3: The effect of number of eigenchannels for JFA with $\mathbf{D}$ trained on data and full JFA. Results on NIST 2006 data.

### 3.3. Effect of zt-norm

The importance of using zt-norm for getting good performance with JFA systems was previously reported in [6, 5]. On contrary, our experience was that omitting zt-norm was not critical for eigenchannel adaptation based system. To verify these contradictory findings, we evaluated both eigenchannel adaptation and full JFA system with and without using zt-norm. As can be seen in Fig. 4, without zt-norm, both eigenchannel adaptation and full JFA system provide very similar performance. However, while only small gain was obtained with zt-norm for eigenchannel adaptation, dramatic improvement was obtained for full JFA system. Note again that gender-dependent zt-norm was used in both cases, which is crucial for good performance even for GI version of full JFA system. With gender-independent zt-norm (results are not shown in the figure), no significant gain was obtained for eigenchannel adaptation [2] and significant degradation in performance was observed for full JFA system compared to the system without zt-norm.

### 3.4. Training eigenchannels for different channel conditions

As described in section 2, our baseline JFA systems were primarily developed for telephone data. All the hyperparameters are trained on telephone data, only 100 additional eigenchannels were trained on microphone data. This strategy was already found to be effective [4] to allow the system to deal with the microphone speech segments. In Fig. 5, results are presented for all four conditions, where enrollment and test seg-

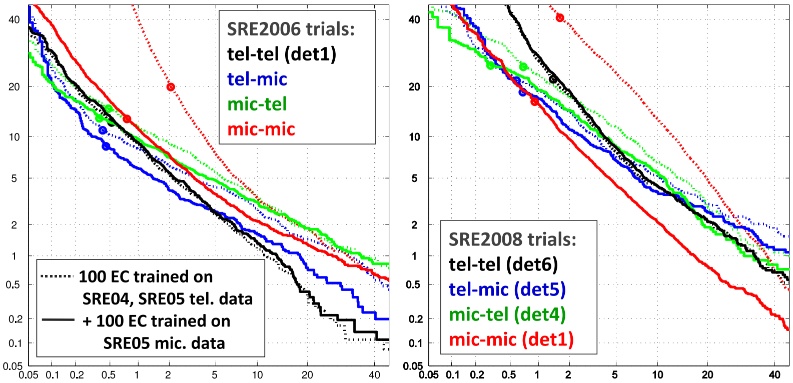


Figure 5: Performance of **MFCC13⇒39** system for different channel conditions.

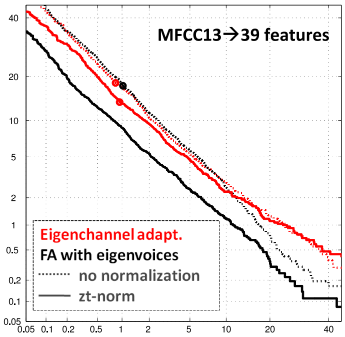


Figure 4: The effect of zt-norm. Results on NIST 2006 data.

ments are recorded either over telephone or microphone. On the left, results are presented for NIST SRE 2006 data described in section 1. On the right, results on corresponding conditions from NIST SRE 2008 evaluations[3] are presented for comparison. The dotted lines represents performance of systems with only 100 eigenchannels trained on SRE04, SRE05 telephone data while systems represented by solid lines make also use of the additional 100 eigenchannels trained also on SRE05 microphone data. We can see that augmenting the original 100 eigenchannels by those trained on microphone data brought negligible degradation for tel-tel condition and large improvement particularly on mic-mic condition. An interesting observation is that, when dropping eigenchannels trained on microphone data, much smaller degradation in performance is obtained for conditions with either enrollment or test segment recorded over telephone compared to the case where both the segments are recoded over microphone.

## 4. Conclusions

In this paper, we have investigated into different variants of JFA used for speaker recognition. We have shown that the full JFA with both eigenchannels and eigenvoices outperforms all simplified variants. The presence of eigenvoices allows for use of increased number of eigenchannels, which would otherwise lead to over-training of the system. We found that **gender-dependent** zt-norm was crucial for good performance of the full JFA system. This suggests, that further conditioning on other dominant speaker characteristics might be beneficial and calls for further investigation.

Although our system was primarily trained on and tuned for telephone data, JFA subsystems can be simply augmented with eigenchannels trained on microphone data, which makes the system performing well also on microphone conditions.

## 5. References

[1] P. Schwarz, P. Matějka and J. Černocký: Hierarchical Structures of Neural Networks for Phoneme Recognition, In Proceedings of ICASSP 2006, May 2006, Toulouse, France

[2] L. Burget, P. Matějka, P. Schwarz, O. Glembek and J. Černocký: Analysis of feature extraction and channel compensation in GMM speaker recognition system, In: IEEE Transactions on Audio, Speech, and Language Processing, Vol. 15, No. 7, 2007, pp. 1979-1986.

[3] N. Brümmer: Spescom DataVoice NIST 2004 system description, in Proc. NIST Speaker Recognition Evaluation 2004, Toledo, Spain, June 2004.

[4] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel: A Study of Inter-Speaker Variability in Speaker Verification, IEEE Transactions on Audio, Speech and Language Processing, July 2008.

[5] Kenny, P., Boulianne, G., Ouellet, P. and P. Dumouchel. Joint factor analysis versus eigenchannels in speaker recognition, In IEEE Transactions on Audio, Speech and Language Processing 15 (4), pp. 1435–1447, May 2007.

[6] R. Vogt, and S. Sridharan: Explicit Modelling of Session Variability for Speaker Verification. Computer Speech & Language 22(1), 2008, pp. 17-38.

[7] L. Burget et al.: BUT system description: NIST SRE 2008, In: Proc. 2008 NIST Speaker Recognition Evaluation Workshop, Montreal, Canada, 2008, `http://www.fit.vutbr.cz/research/view_pub.php?id=8745`

[8] D. Matrouf, N. Sheffer, B. Fauve, and J-F. Bonastre: A Straightforward and Efficient Implementation of the Factor Analysis Model for Speaker Verification, in Proc. ICSLP 2007, Antwerp, Belgium, pp. 1242–1245, August 2007.

[9] D. Reynolds, T. Quatieri, and R. Dunn: Speaker verification using adapted Gaussian mixture models, Digital Signal Processing, vol. 10, pp. 19–41, 2000.

[10] O. Glembek et al.: Comparison of Scoring Methods used in Speaker Recognition with Joint Factor Analysis, In Proc. ICASSP 2009, Taipei, Taiwan, April 2009

[3] `http://www.nist.gov/speech/tests/sre/2008/sre08_evalplan_release4.pdf`

# Analysis of feature extraction and channel compensation in a GMM speaker recognition system

Lukáš Burget*, *Member, IEEE,* Pavel Matějka, *Member, IEEE,* Petr Schwarz, *Member, IEEE,*
Ondřej Glembek, *Student Member, IEEE,* and Jan "Honza" Černocký, *Member, IEEE*

***Abstract*—In this paper, several feature extraction and channel compensation techniques found in state-of-the-art speaker verification systems are analyzed and discussed. For the NIST SRE 2006 submission, Cepstral Mean Subtraction, Feature Warping, RASTA filtering, HLDA, Feature Mapping and Eigenchannel Adaptation were incrementally added to minimize the system's error rate. The paper deals with Eigenchannel Adaptation in more detail, and includes its theoretical background and implementation issues. The key part of the paper is however the post-evaluation analysis, undermining a common myth that "the more boxes in the scheme, the better the system". All results are presented on NIST SRE 2005 and 2006 data.**

***Index Terms*—Speaker recognition, GMM, Feature Warping, RASTA, HLDA, Feature Mapping, Eigenchannel Adaptation.**

**EDICS Category: SPE-SPKR**

## I. Introduction

In the NIST 2006 Speaker Recognition Evaluation [1], the Brno University of Technology (BUT) participated with its own submission and also contributed to systems developed by the STBU[1] consortium. Both the BUT and STBU primary systems were fusions of several individual subsystems, namely: systems based on Gaussian Mixture Modeling (GMM) [2], and systems based on sequence kernel Support Vector Machines (SVM) classifying either GMM mean supervectors [3] or vectors constructed from Maximum Likelihood Linear Regression (MLLR) transformations [4], which are transformations commonly used in speech recognition for speaker adaptation. In this paper, we provide an analysis of the BUT GMM system that took part in both the BUT and STBU primary systems, and which was also submitted as a BUT stand-alone secondary system. The overall description of the BUT and STBU systems can be found in [5], [6].

The BUT GMM system is based on a standard Universal Background Model-Gaussian Mixture Modeling (UBM-GMM) paradigm [2] and employs a number of techniques that have previously proven to improve GMM modeling capability and help fight against the main problem in speaker verification - diversity in channel and acoustic conditions. These techniques are: Cepstral Mean Subtraction, Feature Warping [7], RelAtive SpecTrAl (RASTA) filtering [8], Heteroscedastic Linear Discriminant Analysis (HLDA) [9], Feature Mapping [10] and Eigenchannel Adaptation [11]. The aim of this paper is to analyze the importance of the individual techniques in terms of their contribution to overall system performance.

The paper is organized as follows: A detailed description of the BUT GMM speaker recognition system is provided in section II. Section III documents building the system and reports the improvements in performance obtained by adding individual techniques. Section IV presents our post-evaluation activity and analyzes the importance of the individual techniques in the full system. The result obtained by fusing the GMM system with the SVM-based systems are presented in section V. We conclude the paper in section VI.

## II. System Description

### A. Features

The features used in the system are Mel-frequency cepstral coefficients (13 MFCC coefficients including C0, 20 ms window, 10 ms shift, 23 bands in a Mel filter bank). To compensate for channel mismatch in different conversations, three simple feature processing techniques were successively applied: the cepstral mean over the whole conversation is subtracted from the features, Feature Warping [7] (3 sec window, warping into a normal distribution) is applied and finally temporal trajectories of individual feature vector coefficients are filtered using a standard RASTA filter [8][2]. After this processing, each feature vector is augmented with its first, second and third order derivatives. This results in 52 dimensional feature vectors containing information about the context of 13 frames.

### B. Segmentation

At this stage, non-speech frames are discarded and only speech frames are considered in the following stages of training models and verification. Speech/non-speech segmentation is performed by our Hungarian phoneme recognizer [12],

All authors are with Speech@FIT, Faculty of Information Technology, Brno University of Technology, Czech Republic. Lukáš Burget is the corresponding author.

This work was partly supported by by European projects AMIDA (IST-033812) and Caretaker (FP6-027231), by Grant Agency of the Czech Republic under projects No. 102/05/0278 and GP102/06/383 and by the Czech Ministry of Education under project No. MSM0021630528. The hardware used in this work was partially provided by CESNET under projects No. 119/2004, No. 162/2005 and No. 201/2006.

We are grateful to the anonymous reviewers for their valuable comments on the first version of this paper. Many thanks to Ken Froehling from the Faculty of Electrical Engineering and Communication of BUT for help with the English of the paper.

[1] BUT, TNO Human Factors (The Netherlands), Spescom DataVoice (South Africa) and the University of Stellenbosch (South Africa).

[2] Cepstral Mean Subtraction has no effect after the application of Feature Warping and RASTA filtering as both techniques also ensure the mean removal. However, it will be interesting to see the effectiveness of these techniques compared to Cepstral Mean Subtraction alone.

where all phoneme classes are linked to speech classes. A postprocessing with two rules based on the short time energy of the signal is applied: 1) If the average energy in a speech segment is 30dB less than the maximum energy in the conversation side, then the segment is labeled as silence. 2) If the energy in the opposite conversation side[3] is bigger than the maximum energy minus 3dB in the processed side, the segment is also labeled as silence.

### C. HLDA

As the next step, we have employed Heteroscedastic Linear Discriminant Analysis (HLDA), which is also in common use in speech recognition systems. HLDA provides a linear transformation that can de-correlate the features and reduce the dimensionality while preserving the discriminative power of features. The theory of HLDA is described in detail in [9], [13]. HLDA needs classes to estimate its class-covariance statistics (which are then used to estimate the transformation matrix). For this purpose, GMM with 2048 Gaussian components is trained on test data from SRE2004 and the feature frames aligned with individual GMM mixture components are considered as classes. HLDA transformation reducing the dimensionality from 52 to 39 is estimated. GMM is then updated in the new HLDA space (by projecting collected class-covariance and mean statistics through HLDA transformation). Features are also projected into HLDA space and GMM is re-estimated (still only on SRE2004 test data) by few additional Expectation-Maximization (EM) iterations to obtain the Universal Background Model (UBM).

### D. Feature Mapping

To further compensate for channel mismatch, Feature Mapping [10] was applied to all enrollment and test conversations. Feature Mapping requires a set of models, each adapted from UBM using data of particular acoustic condition (channel). We have used 14 such models: 6 models were adapted for 3 channels (cell,cord,stnd) and 2 genders given the labels from 2004 test data. The remaining 8 models were initially adapted for 4 channels (cdma, cord, elec, gsmc) and 2 genders using the TNO Feature Mapping labels used in SRE-2005. However, these 8 models were then iteratively used to re-cluster the training data in an unsupervised fashion and again adapted using the new clustering (20 iterations lead to stable clustering) [14].

### E. Training speaker model and verification

Each speaker model is obtained by a traditional *relevance Maximum A-Posteriori (MAP)* adaptation [15] of UBM using enrollment conversation. Only means are adapted with a relevance factor $\tau = 19$.

In the verification phase, standard Top-N Expected Log Likelihood Ratio (ELLR) scoring [15] is used to obtain a verification score, where $N = 10$ in our system. However, for each trial, both the speaker model and UBM are adapted to the channel of test conversation using simple Eigenchannel Adaptation [11] prior to computing the log likelihood ratio score. Note, that when T-norm [16] is used to normalize the score, each T-norm model is also adapted to the channel of relevant tested conversation.

### F. Eigenchannel subspace estimation

We adopted the term 'eigenchannel' as used in speaker recognition from Kenny [17]. It was introduced to the NIST SRE by SDV in 2004 [11], revisited by Kenny and Vogt [18] in SRE 2005, and again by several sites in various forms in SRE 2006.

Let *supervector* be a $MD$ dimensional vector constructed by concatenating all GMM mean vectors and *normalized by corresponding standard deviations*. $M$ is the number if Gaussian mixture components in GMM and $D$ is dimensionality of features. Before Eigenchannel Adaptation can be applied, we must identify directions in which the *supervector* is mostly affected by a changing channel. These directions, which we will refer to as eigenchannels, are defined by columns of $MD \times R$ matrix $\mathbf{V}$, where $R$ is the chosen number of eigenchannels ($R = 30$ in our system). The matrix $\mathbf{V}$ is given by $R$ eigenvectors of average within class covariance matrix, where each class is represented by supervectors estimated on different segments spoken by the same speaker.

More precisely, we have selected all (310) speakers from NIST SRE2004 data for which at least two conversations are available. For each speaker, $i$, and all his conversations, $j = 1, \ldots, J_i$, UBM is adapted to obtain a supervector, $\mathbf{s}_{ij}$. The corresponding speaker average supervector given by $\overline{\mathbf{s}}_i = \sum_{j=1}^{J_i} \mathbf{s}_{ij}/J_i$ is subtracted from each supervector, $\mathbf{s}_{ij}$, and resulting vectors form columns of $MD \times J$ matrix $\mathbf{S}$, where $J$ is the number of all conversations from all selected speakers ($J = 2961$ in our case). Eigenchannels (columns of matrix $\mathbf{V}$) are given by $R$ eigenvectors of $MD \times MD$ average within speaker covariance matrix[4] $\frac{1}{J}\mathbf{S}\mathbf{S}^T$ corresponding to $R$ largest eigenvalues. Unfortunately, for our system, where $MD = 2048 \times 39 = 79872$, direct computation of these eigenvectors is unfeasible. A possible solution is to compute eigenvectors, $\mathbf{V}'$, of $J \times J$ matrix $\frac{1}{J}\mathbf{S}^T\mathbf{S}$; eigenchannels are then given by $\mathbf{V} = \mathbf{S}\mathbf{V}'$. In case the maximum a-posteriori (MAP) criterion is used for Eigenchannel adaptation (see below), the length of each eigenchannel must be also normalized to the average within speaker standard deviation of supervectors along the direction of the eigenchannel (i.e. each eigenvector obtained in the previous step must be multiplied by the square root of the corresponding eigenvalue). This normalization is irrelevant in the case of maximum likelihood (ML) criterion.

### G. Eigenchannel Adaptation

Once the eigenchannels are identified, a speaker model (or UBM) can be adapted to the channel of a test conversation by shifting its supervector in the directions given by eigenchannels to better fit the test conversation data. Mathematically,

[3] In NIST SRE2006 evaluations, our system participated only in the primary condition, where two separate recordings for the two sides of each phone conversation are available.

[4] Note that matrix $\frac{1}{J}\mathbf{S}\mathbf{S}^T$ is a true covariance matrix as the zero mean over columns of $S$ is guaranteed by the subtraction of the speaker average supervectors described above.

this can be expressed as finding the *channel factors*, $\mathbf{x}$, that maximize the following MAP criterion:

$$p(\mathbf{O}|\mathbf{s}+\mathbf{V}\mathbf{x})\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{I}),\qquad(1)$$

where $\mathbf{s}$ is a supervector representing the model to be adapted[5], $p(\mathbf{O}|\mathbf{s}+\mathbf{V}\mathbf{x})$ is the likelihood of the test conversation given the adapted supervector (model) and $\mathcal{N}(\cdot;\mathbf{0},\mathbf{I})$ denotes a normally distributed vector. Assuming a fixed occupation of the Gaussian mixture components by test conversation frames, $\mathbf{o}_t, t = 1,\ldots,T$, it can be shown [11] that $\mathbf{x}$ maximizing criterion (1) is given by:

$$\mathbf{x} = \mathbf{A}^{-1}\sum_{m=1}^{M}\mathbf{V}_m^T\sum_{t=1}^{T}\gamma_m(t)\frac{\mathbf{o}_t-\boldsymbol{\mu}_m}{\boldsymbol{\sigma}_m},\qquad(2)$$

where $\mathbf{V}_m$ is $M\times R$ part of matrix $\mathbf{V}$ corresponding to the $m^{th}$ mixture component, $\gamma_m(t)$ is the probability of occupation mixture component $m$ at time $t$, $\boldsymbol{\mu}_m$ and $\boldsymbol{\sigma}_m$ are the mixture component's mean and standard deviation vectors and

$$\mathbf{A} = \mathbf{I} + \sum_{m=1}^{M}\mathbf{V}_m^T\mathbf{V}_m\sum_{t=1}^{T}\gamma_i(t).\qquad(3)$$

In our implementation, occupation probabilities, $\gamma_m(t)$, are computed using UBM and assumed to be fixed for given test conversation. This allows us to pre-compute matrix $\mathbf{A}^{-1}$ only once for each test conversation. For each frame, only Top-N occupation probabilities are assumed not to be zero. In the following ELLR scoring, only the same top-N mixture components are also considered. All these facts ensure that adapting and scoring different speaker or T-norm models on a test conversation can be performed very efficiently.

Eigenchannel Adaptation can be also performed by maximizing ML criterion instead of MAP criterion. This corresponds to dropping the prior term, $\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{I})$, in criterion (1) and term $\mathbf{I}$ in equation 3. In our experiments, there is always enough adaptation data (test conversations contain approximately 2.5 minutes of speech) making the prior term in MAP criterion negligible. Therefore, we have not found any differences in performance when using the two criteria.

Our system uses a very simple scheme of modeling channel variability that affects only the verification phase. However, more sophisticated schemes can be considered. In [19] the verification phase is equivalent to that described here, however, modeling channel variability is considered also in training speaker models. This may become important especially when speaker models are trained using more than one enrollment conversation.

A very elaborate scheme can be found in [17], where modeling channel variability is considered in all phases: training background model, training speaker models and verification. Instead of finding eigenvectors, channel subspace $\mathbf{V}$ is obtained also by maximizing MAP criterion similar to (1). For enrollment data, instead of finding MAP point estimates of model parameters, posterior probabilities of model parameters are considered and integrated over to obtain the likelihood score for a test conversation.

## III. Building the system

In the following experiments, results will be presented for "1-side training, 1-side test, all trials" condition from SRE2005 NIST evaluation, which we have used for system development, and for primary condition (1-side training, 1-side test, English only trials) from SRE2006 NIST evaluation. In the tables, results are presented in terms of EER (Equal Error Rate) and $C_{\mathrm{Det}}^{\mathrm{min}}$ as defined by SRE2006 NIST evaluation rules [1]. For SRE2006 primary condition, performances are also presented in the form of DET (Detection Error Tradeoff) curves.

Table I and figure 1 document the process of building our system. It shows line-by-line the improvements in performance obtained by successively adding different techniques. Our starting point was GMM system with 2048 Gaussian mixture components, features were 13 MFCC coefficients augmented with their deltas and processed by cepstral mean subtraction. The error rate of this system is very high and is almost halved by simply adding RASTA filtering. Replacing RASTA with Feature Warping improved the performance; however, a further small gain was obtained from the combination of both techniques. The application of RASTA filtering on top of Feature Warping appeared to be slightly more advantageous than doing it in the opposite order. In the next two steps, features were also augmented with double-delta and triple-delta coefficients. While adding double-deltas is clearly beneficial for both SRE2005 and SRE2006 evaluation sets, the advantage of adding triple-deltas, which we have seen during development on SRE2005 data, was not confirmed on SRE2006.

The following three steps, each significantly improving the system performance, were: projection of 52 dimensional features into 39 dimensional HLDA space, application of our 14 classes Feature Mapping and Eigenchannel Adaptation.

So far, all the presented results were obtained without normalizing the verification scores by any standard technique, such as T-normalization or Z-normalization (Z-norm/T-norm) [16]. As can be seen in Table I, T-norm was not effective in improving the performance of our full system. We have also experimented with Z-norm and ZT-norm, nevertheless, results obtained with all normalization techniques were mixed and unconvincing. This contradicted the conclusions drawn in [17], [18], [19], where Z-norm or ZT-norm was found necessary for making channel variability modeling techniques really effective.

Most of GMM based speaker verification systems, for which the results are published by various sites, use less than 2048 Gaussian components. The last line of table I show results for a system with the usual number of only 512 Gaussian components, which is otherwise identical to our full system. It can be seen that the performance of a 2048 component system is superior to this smaller one.

## IV. Post-evaluation analysis

In the previous section, we have shown how adding individual techniques improves system performance. However, it will be even more interesting to see whether and how the individual techniques are important in the full system.

[5] Note again that by our definition, a supervector is a mean supervector normalized by the corresponding standard deviations.

| System | SRE2005 | | SRE2006 | |
|---|---|---|---|---|
| | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ |
| MFCC+Δ, CMS, 2048 G. | 26.6% | .089 | 23.8% | .088 |
| + RASTA | 14.3% | .055 | 11.8% | .059 |
| + Feature Warping | 12.4% | .052 | 10.0% | .051 |
| + ΔΔ | 11.2% | .047 | 9.1% | .049 |
| + ΔΔΔ | 10.6% | .047 | 9.3% | .048 |
| + HLDA (52→39) | 9.7% | .042 | 8.2% | .041 |
| + Feature Mapping | 7.3% | .033 | 6.2% | .032 |
| + Eigenchannel Adapt. | 4.6% | .020 | 4.0% | .020 |
| + T-norm | 4.6% | .020 | 4.0% | .018 |
| Full system, 512 Gauss. | 4.9% | .026 | 4.7% | .024 |

TABLE I
THE IMPROVEMENTS IN PERFORMANCE OBTAINED BY SUCCESSIVELY ADDING DIFFERENT TECHNIQUES.

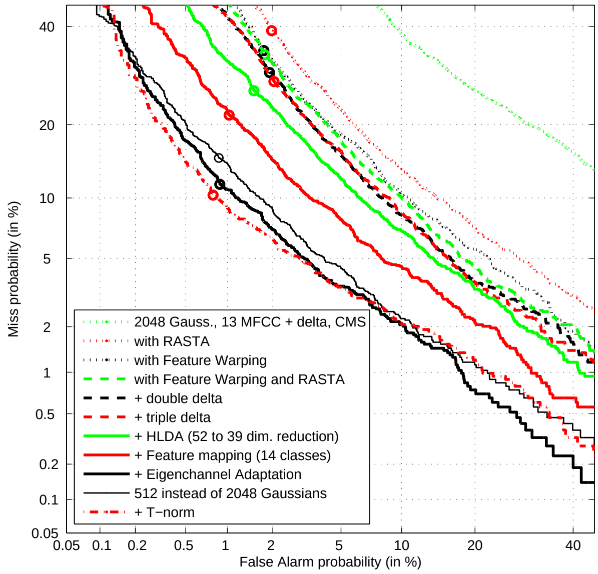


Fig. 1. DET curves showing improvement in successive adding different techniques.

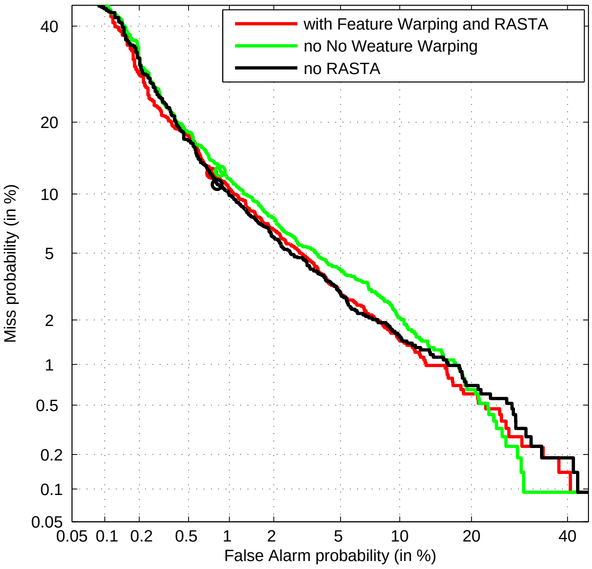


Fig. 2. The importance of RASTA filtering and Feature Warping.

| System | SRE2005 | | SRE2006 | |
|---|---|---|---|---|
| | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ |
| Full system | 4.5% | .019 | 3.8% | .020 |
| No RASTA | 4.4% | .019 | 3.8% | .019 |
| No Feature Warping | 5.1% | .020 | 4.3% | .021 |

TABLE II
THE IMPORTANCE OF RASTA AND FEATURE WARPING.

### A. The importance of RASTA and Feature Warping

Table II and figure 2 present results obtained with the baseline full system[6] and two of its modifications leaving out either RASTA filtering or Feature Warping. While Feature Warping turns out to be an important part of the system, leaving out RASTA filtering even slightly improves the system performance. This may support the conclusions in [8], where RASTA was found to discard important speaker information lying under its cut-off frequency and a filter more appropriate for speaker verification was designed.

### B. Analyzing the effect of HLDA

The left half of Table III shows the effect of HLDA for systems without the following Feature Mapping and Eigenchannel Adaptation. The first two results (already presented in Table I) demonstrate the effectiveness of HLDA at this stage. The dimensionality reduction from 52 to 39 was chosen as exactly the same scheme had already been proven to be effective for speech recognition [20]. Since it was not clear whether this scheme is optimal for our speaker verification system, reductions to various dimensionalities were examined and the best results were obtained without any dimensionality reduction[7] (last line of Table III).

The situation is different in the right half of Table III, where Feature Mapping and Eigenchannel Adaptation are used. Performances of systems using HLDA are still superior to the one that leaves HLDA out; however, the system with dimensionality reduction outperforms the one without reduction. The possible explanation is that the significant increase in GMM (and supervector) size makes it impossible to robustly estimate eigenchannels given the limited number of supervectors available for their estimation. The summary of HLDA and MLLT results can be also found in figure 3.

### C. Eigenchannels vs. Feature Mapping

The left half of Table IV and dotted DET curves in figure 4 show the effect of Feature Mapping for systems without the following Eigenchannel Adaptation. The first two results (already presented in Table I) demonstrate the effectiveness

[6] System with 2 Gender Feature Mapping (see below) is used as a baseline system in this experiment for efficiency reasons.

[7] HLDA without dimensionality reduction is often referred to as a Maximum Likelihood Linear Transform (MLLT) [21]

| System | Without channel comp. | | | | With channel comp. | | | |
|---|---|---|---|---|---|---|---|---|
| | SRE2005 | | SRE2006 | | SRE2005 | | SRE2006 | |
| | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ |
| No HLDA | 10.6% | .047 | 9.3% | .048 | 5.1% | .024 | 5.0% | .025 |
| HLDA 52 → 39 | 9.7% | .042 | 8.2% | .041 | 4.5% | .019 | 3.8% | .020 |
| HLDA 52 → 52 | 8.7% | .038 | 7.5% | .037 | 4.6% | .023 | 4.2% | .021 |

TABLE III
THE EFFECT OF HLDA ON SYSTEM PERFORMANCE.

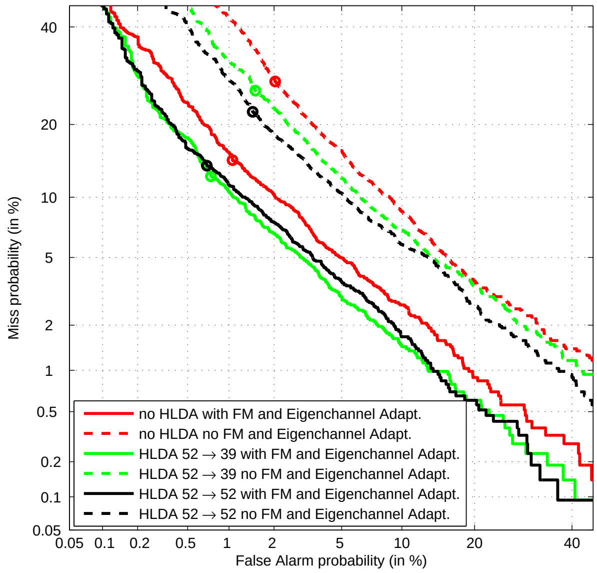


Fig. 3. The effect of HLDA and MLLT (HLDA without dimensionality reduction) on system performance.

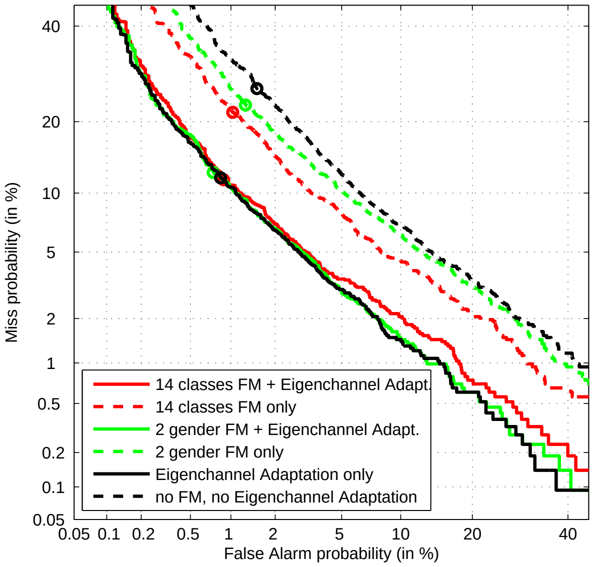


Fig. 4. The importance of Feature Mapping and Eigenchannel Adaptation.

of Feature Mapping at this stage. In the third line, the performance of a system using Feature Mapping based on only two models adapted on male and female specific data is shown. This allows us to compensate for the fact that our system uses only a single UBM instead of the usual approach where two genders are handled separately using two UBMs. Although such 2-gender Feature Mapping significantly outperforms the system leaving Feature Mapping out, it still reaches only about half of the gain in performance compared to 14 classes Feature Mapping used in our final system.

The right half of Table IV and solid DET curves in figure 4 show similar results for systems applying also Eigenchannel Adaptation. We can see that without Feature Mapping, Eigenchannel Adaptation causes an impressive improvement in system performance (more than 50% relative in both EER and $C_{\mathrm{Det}}^{\mathrm{min}}$ points). There is *no advantage in using Feature Mapping after the Eigenchannel Adaptation is applied*, which allows us to simplify the verification system considerably by leaving Feature Mapping out. In fact, the use of our 14 classes Feature Mapping causes even slight degradation in the performance. It was surprising for us that even 2-gender Feature Mapping did not turn out to be effective, as eigenchannels are not trained to model the directions of differences between male and female specific models.

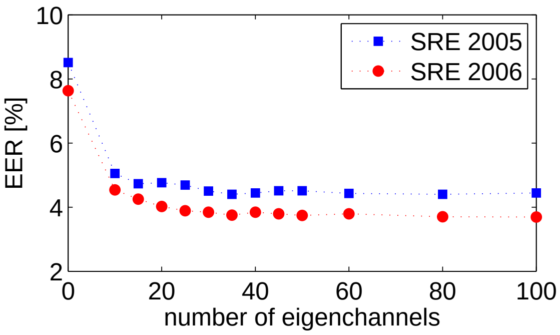


Fig. 5. The dependency of EER on the number of eigenchannels used for adaptation.

## D. Number of eigenchannels

The number of eigenchannels was chosen to be $R = 30$ for our system submitted to SRE2006 NIST evaluations. Figure 5 shows the dependency of EER on the number of eigenchannels used for adaptation. A similar trend has also been observed for $C_{\mathrm{Det}}^{\mathrm{min}}$ values. It can be seen that our system is not very sensitive to the exact selection of the number of eigenchannels.

| System | Without Eigenchannel Adapt. | | | | With Eigenchannel Adapt. | | | |
|---|---|---|---|---|---|---|---|---|
| | SRE2005 | | SRE2006 | | SRE2005 | | SRE2006 | |
| | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ |
| No Feature Mapping | 9.7% | .042 | 8.2% | .041 | 4.6% | .019 | 3.8% | .020 |
| 14 classes Feature Mapping | 7.3% | .033 | 6.2% | .032 | 4.6% | .020 | 4.0% | .020 |
| 2-gender Feature Mapping | 8.5% | .037 | 7.6% | .036 | 4.5% | .019 | 3.8% | .020 |

TABLE IV
THE IMPORTANCE OF FEATURE MAPPING AND EIGENCHANNEL ADAPTATION.

| System | SRE2005 | | SRE2006 | |
|---|---|---|---|---|
| | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ | EER | $C_{\mathrm{Det}}^{\mathrm{min}}$ |
| no T-n., no RASTA, no FM, 50 EA | 4.4% | .017 | 3.6% | .018 |

TABLE V
RESULTS OF THE FINAL TUNED AND SIMPLIFIED SYSTEM.

## V. FUSING WITH SVM BASED SYSTEMS

The performance of the GMM system was also tested in combination with speaker recognition systems based on a different classification paradigm – Support Vector Machines (SVM). Figure 6 contains a summary of results for SRE2006 primary condition. Results are presented for BUT stand-alone systems as well as for fused systems that were BUT and STBU submissions into the SRE2006 NIST evaluations.

These systems (from the worst to the best) are:

- SVM-MLLR, where MLLR and constrained MLLR (CMLLR) speaker adaptation matrices from a speech recognition system are classified by SVM. Two variants are shown: with and without T-norm
- SVM-GMM, where GMM supervectors are classified by SVMs. Two variants are shown: with and without T-norm
- **GMM** is the full system described in this paper. Two variants (already presented in Table I) are shown: with and without T-norm
- BUT02 is a fusion of 3 systems: GMM, SVM-GMM and SVM-MLLR, all with T-norm applied
- BUT01 (BUT primary system) is a fusion of 6 systems: GMM, SVM-GMM and SVM-MLLR, each in two variants: with and without T-norm
- STBU1-N is fusion of 10 systems from the partners in the STBU consortium.
- STBU1-U (STBU primary system) is fusion of the same 10 systems, plus one more SVM-GMM system implementing unsupervised adaptation to test data according to SRE2006 NIST evaluation rules [1].

A detailed description of different systems can be found in [5], [6]. The fusion was performed using linear logistic regression implemented in the FoCal toolkit[8] and it is also described and commented on in [6].

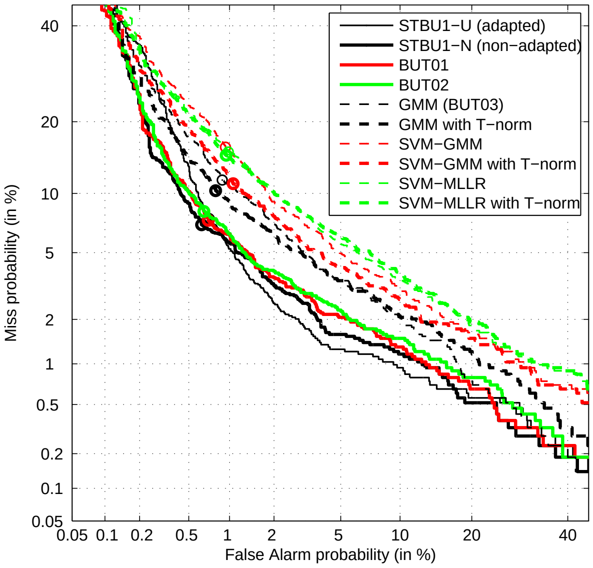


Fig. 6. Fusion of GMM system with SVM based systems.

## VI. CONCLUSION

BUT GMM system contains nothing more than techniques that were already published – its main contribution is in a thorough analysis and discussion of these techniques in a full speaker recognition system. Starting in the feature extraction, the main conclusion is that RASTA did not help in the full system. On the other hand, HLDA significantly improved its performances, although we know that there is still work to be done (different dimensionality reductions examined with the full system, not using triple-deltas, etc.). In fighting the channel variability, even the simple Eigenchannel Adaptation turned out to be very effective, erasing the advantages of Feature Mapping, which is actually not important when applied together with Eigenchannel Adaptation. Table V presents the results of the final tuned and simplified system, containing 50 eigenchannels, no T-norm, no RASTA and no Feature Mapping. All the conclusions may, however, not hold for other than 1-side training, 1-side test condition examined in this work. Our current and future work aims at these conditions as well as at using the described GMM system as an excellent baseline for further experiments.

[8] www.dsp.sun.ac.za/~nbrummer/focal/

## REFERENCES

[1] "The NIST year 2006 speaker recognition evaluation plan," 2006, Available from: http://www.nist.gov/speech/tests/spk/2006/sre-06_evalplan-v9.pdf.

[2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1–3, pp. 19–41, 2000.

[3] A. Solomonoff, W. Campbell, and I. BoardmanCampbell, "Advances in channel compensation for SVM speaker recognition," in *Proc. ICASSP*, vol. I, Philadelphia, PA, USA, Mar. 2005, pp. 629–632.

[4] A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, “MLLR transforms as features in speaker recognition,” in *Proc. Eurospeech*, Lisbon, Portugal, Sep. 2005, pp. 2425–2428.

[5] P. Matějka, L. Burget, P. Schwarz, O. Glembek, M. Karafiát, F. Grézl, J. Černocký, D. A. van Leeuwen, N. Brümmer, and A. Strasheim, “STBU system for the NIST 2006 speaker recognition evaluation,” in *Proc. ICASSP*, Honolulu, Hawaii, USA, Apr. 2007.

[6] N. Brümmer, L. Burget, J. Černocký, O. Glembek, F. Grézl, M. Karafiát, D. A. van Leeuwen, P. Matějka, P. Schwarz, and A. Strasheim, “Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006,” *IEEE Trans. on Audio, Speech and Language Processing*, 2007, submitted.

[7] J. Pelecanos and S. Sridharan, “Feature warping for robust speaker verification,” in *Proc. A Speaker Odyssey*, Crete, Grece, 2001, pp. 213–218.

[8] S. van Vuuren and H. Hermansky, “On the importance of components of the modulation spectrum for speaker verification,” in *Proc. ICSLP*, vol. 7, Sydney, Australia, May 1998, pp. 3205–3208.

[9] N. Kumar, “Investigation of silicon-auditory models and generalization of linear discriminant analysis for improved speech recognition,” Ph.D. dissertation, John Hopkins University, Baltimore, 1997.

[10] D. A. Reynolds, “Channel robust speaker verification via feature mapping,” in *Proc. ICASSP*, vol. II, Apr. 2003, pp. 53–56.

[11] N. Brümmer, “Spescom DataVoice NIST 2004 system description,” in *Proc. NIST Speaker Recognition Evaluation 2004*, Toledo, Spain, Jun. 2004.

[12] P. Schwarz, P. Matějka, and J. Černocký, “Hierarchical structures of neural networks for phoneme recognition,” in *Proc. ICASSP*, France, May 2006, pp. 325–328.

[13] M. J. F. Gales, “Semi-tied covariance matrices for hidden markov models,” *IEEE Transactions Speech and Audio Processing*, vol. 7, pp. 272–281, 1999.

[14] M. Mason, R. Vogt, B. Baker, and S. Sridharan, “Data-driven clustering for blind feature mapping in speaker verification,” in *Proc. Eurospeech*, Lisbon, Portugal, Sep. 2005, pp. 3109–3112.

[15] D. A. Reynolds, “Comparison of background normalization methods for text-independent speaker verification,” in *Proc. Eurospeech*, Rhodes, Greece, Sep. 1997, pp. 963–966.

[16] R. Auckenthaler, M. Carey, and H. Lloyd-Tomas, “Score normalization for text-independent speaker verification systems,” *Digital Signal Processing*, vol. 10, pp. 42–54, 2000.

[17] P. Kenny and P. Dumouchel, “Disentangling speaker and channel effects in speaker verification,” in *Proc. ICASSP*, vol. 1, Montreal, Canada, May 2004, pp. 47–40.

[18] R. Vogt, B. Baker, and S. Sridharan, “Modelling session variability in text-independent speaker verication,” in *Proc. Eurospeech*, Lisbon, Portugal, Sep. 2005, pp. 3117–3120.

[19] R. Vogt and S. Sridharan, “Experiments in session variability modelling for speaker verification,” in *Proc. ICASSP*, vol. 1, Toulouse, France, May 2006, pp. 897–900.

[20] T. Hain, L. Burget, J. Dines, G. Garau, M. Karafiat, M. Lincoln, I. McCowan, D. Moore, V. Wan, R. Ordelman, and S. Renals, “The 2005 AMI system for the transcription of speech in meetings,” in *Proc. NIST Rich Transcription 2005 Spring Meeting Recognition Evaluation*, Edinburgh,UK, Jul. 2005.

[21] R. Gopinath, “Maximum likelihood modeling with Gaussian distributions for classification,” in *Proc. ICASSP*, vol. II, Seattle, Washington, USA, May 1998, pp. 661–664.

**Lukáš Burget** (Ing. [MS]. Brno University of Technology, 1999, Ph.D. Brno University of Technology, 2004) is employed as an assistant professor at the Faculty of Information Technology, University of Technology, Brno, Czech Republic. The topic of his PhD dissertation that he successfully defended in November 2004 was: ”Complementarity of Speech Recognition Systems and System Combination”. From 2000 to 2002, he was a visiting researcher at OGI Portland, USA under the supervision of Prof. Hynek Hermansky. He is a member of IEEE and ISCA. His scientific interests are in the field of speech processing, namely acoustic modeling for speech recognition.

**Pavel Matějka** (Ing. [MS]. Brno University of Technology, 2001) is a PhD student at the Institute of Radio-electronics, Faculty of Electrical Engineering and Communication and Department of Computer Graphics and Multimedia, FIT, BUT. He is planning to submit his doctoral thesis “Language identification based on phonetic cues” in summer 2007. He has been with the Anthropic speech processing group at the Oregon Graduate Institute of Science and Technology, USA. He is a member of IEEE and ISCA. His research interests include speaker recognition, language identification, speech recognition, namely phoneme recognition based on novel feature extractions (temporal patterns) and neural networks. He has been active also in keyword-spotting and on-line implementation of speech processing algorithms. He was a finalist in the Student paper contest at ICASSP2006 in Toulouse.

**Petr Schwarz** (Ing. [MS]. Brno University of Technology, 2001) is a PhD student of Speech processing group at the Faculty of Information Technology (FIT), BUT since September 2001 and is planning to submit his doctoral thesis “Robust phoneme recognition” in 2007. He has been with the Anthropic speech processing group of the Oregon Graduate Institute of Science and Technology, USA. He is a member of IEEE and ISCA. His research interests include speech recognition, namely phoneme recognition based on novel feature extractions (temporal patterns) and neural networks. He has been active also in keyword-spotting and on-line implementation of speech processing algorithms.

**Ondřej Glembek** (Ing. [MS]. Brno University of Technology, 2005) was student at the Brno University of Technology, Faculty of Electrical Engineering and Computer, later the Faculty of Information Technology from 1999. From September till December 2003, he was at the University of Joensuu, Finland as a participant of the Socrates/Erasmus program. From October till November 2004, he was working on a project concerning wavelet transforms at Izhevsk State Technical University, Izhevsk, Russia. From 2005, he is a PhD student in Speech@FIT - he is concentrating on acoustic modeling for speech recognition, recognition of Czech and STK toolkit development.

**Jan “Honza” Černocký** (Ing. [MS] 1993 Brno University of Technology (BUT); Dr. [PhD] 1998 Universite Paris XI and BUT) was with the Institute of Radio-electronics, BUT (Faculty of Electrical Engineering and Computer Science) as an assistant professor from 1997. Since February 2002, he is with the Faculty of Information Technology (FIT), BUT as an Associate Professor (Doc.) and Deputy Head of the Institute of Computer Graphics and Multimedia. With Prof. Hynek Hermansky he is leading the Speech@FIT group at FIT BUT. He supervises several PhD students, and coordinates Speech@FIT activities in several European and national projects. His research interests include signal processing, speech processing (very low bit rate coding, verification, recognition), segmental methods, data-driven determination of speech units and speech corpora. He is a member of IEEE and ISCA and serves on the board of the Czechoslovak section of IEEE.

# Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006

Niko Brümmer, Lukáš Burget, *Member, IEEE,* Jan “Honza” Černocký *Member, IEEE,*
Ondřej Glembek, *Student Member, IEEE,* František Grézl, *Member, IEEE,* Martin Karafiát, *Member, IEEE,*
David A. van Leeuwen, Pavel Matějka, *Member, IEEE,* Petr Schwarz, *Member, IEEE,*
Albert Strasheim

***Abstract*—This paper describes and discusses the ‘STBU’ speaker recognition system, which performed well in the NIST Speaker Recognition Evaluation 2006 (SRE). STBU is a consortium of 4 partners: Spescom DataVoice (South Africa), TNO (The Netherlands), BUT (Czech Republic) and University of Stellenbosch (South Africa). The STBU system was a combination of three main kinds of sub-systems: (1) GMM, with short-time MFCC or PLP features, (2) GMM-SVM, using GMM mean supervectors as input to an SVM, and (3) MLLR-SVM, using MLLR speaker adaptation coefficients derived from an English LVCSR system. All sub-systems made use of supervector subspace channel compensation methods—either eigenchannel adaptation or nuisance attribute projection. We document the design and performance of all sub-systems, as well as their fusion and calibration via logistic regression. Finally, we also present a cross-site fusion that was done with several additional systems from other NIST SRE-2006 participants.**

***Index Terms*—Speaker recognition, GMM, SVM, eigenchannel, NAP, Fusion.**

## I. Introduction

This paper documents significant elements of the state-of-the-art in text-independent telephone speaker recognition, as measured in the NIST Speaker Recognition Evaluation 2006 (SRE), via a description of the design and performance of the ‘STBU’ submission. It expands on a short paper published at ICASSP [1]. The U.S. National Institute of Standards and Technology (NIST) organizes yearly SRE evaluations [2], [3] to contribute to the direction of research efforts and to calibrate the technical capabilities of different academic and industrial sites active in text-independent speaker recognition.

The STBU submission to the NIST SRE-2006 was the result of a collaboration between four institutes:

- Spescom DataVoice (SDV), South Africa,
- TNO, The Netherlands,
- Brno University of Technology (BUT), Czech Republic, and
- University of Stellenbosch (SUN), South Africa.

The STBU consortium was formed to learn and share the technologies and available know-how among partners. Another, equally important, reason to join efforts was that most successful submissions to NIST evaluations fuse the results of several sub-systems to decrease error rates. Simply put, a consortium can generate more diverse systems, and even if the theoretical base is very similar, subtle details in implementation, features, background models, channel normalization and training can make the fused system more accurate.

Admittedly, this paper is not for novices in speaker recognition. Rather, it assumes familiarity with basic approaches such as Universal Background Model-Gaussian Mixture Modelling (UBM-GMM) [4], sequence kernel Support Vector Machines [5] and more advanced channel compensation approaches such as Eigenchannel Adaptation [6] and Nuisance Attribute Projection (NAP) [7]. Further, the reader is assumed to be familiar with the NIST SRE-2006 task of speaker detection [8] and specifically with the ‘1conv4w-1conv4w’ condition[1], where a *detection trial* consists of a pair of speech segments, and where the objective of the exercise is to decide independently for each of several thousand trials, whether the two segments were spoken by the same speaker, or by two different speakers. *Speech segment* here denotes an excerpt of approximately 5 minutes, from one of the 2 channels of a 4-wire recording of a telephone conversation between two people.

The paper is organized as follows: Section II presents the basic system types grouped into three categories. Section III presents the systems from different STBU sites in more detail. In Section IV, we describe in detail the theory and implementation of system fusion and calibration using logistic regression. In particular, we discuss how calibration was done to meet both the traditional $C_{\text{det}}$ and the new $C_{\text{llr}}$ metrics. Results are presented in Section V—this section also analyzes language dependence which was an important issue in SRE-2006. Finally, Section VI presents a cross-site fusion of STBU

The authors appear in alphabetical order

Niko is with Spescom DataVoice, Stellenbosch, South Africa and with University of Stellenbosch.

Pavel, Lukáš, Petr, Ondřej, Martin, František and Honza are with Speech@FIT, Faculty of Information Technology Brno University of Technology, Czech Republic.

David is with TNO Human Factors, Postbus 23, 3769 ZG Soesterberg, The Netherlands.

Albert is with University of Stellenbosch, Department of Electrical and Electronic Engineering, Stellenbosch, South Africa.

[1] For details see the evaluation plan, via http://www.nist.gov/speech/tests/spk/2006/

sub-systems together with several systems from other SRE-2006 participants. We conclude the paper in Section VII.

## II. System description

We used three basic system types: Eigenchannel GMM, GMM-SVM and MLLR-SVM. All sub-systems had in common that they used one of two forms of linear supervector subspace channel compensation technique: (i) For *eigenchannel adaptation*, supervectors were extracted from GMM models, compensated for channel effects, translated back to adapted GMM models and then employed in the usual way to score the tests. (ii) In the case of the SVM-based systems, supervectors were extracted either from GMMs or from MLLR adaptation coefficients and were then subjected to *nuisance attribute projection* to cancel channel effects. Following that, the supervectors are employed in the usual way to train SVM models which can be scored against test supervectors. More detail follows below.

### A. Common signal processing

All sub-systems used standard features such as Mel-frequency cepstral coefficients (MFCC) or perceptual linear prediction (PLP) features. The basic cepstral features were augmented with derivatives up to third order. A set of several frame selection criteria were applied: (a) frame energy must be more than than 30 dB below the maximum frame energy; (b) frame energy at least 3 dB above energy in other channel (cross channel squelch); (c) segmentation from BUT's Hungarian phone recognizer; (d) strongly voiced syllable nuclei detector; (e) ASR word transcript segmentation provided by NIST. RASTA (relative spectral) filtering [9], short-time Gaussianization [10] and heteroscedastic linear discriminant transformation (HLDA) [11], [12] were used for basic channel normalization, feature decorrelation and dimensionality reduction.

### B. Feature mapping

TNO and BUT used the channel-compensation technique of feature mapping [13] to post-process all of their acoustic features. However, post-evaluation experiments by BUT [14] strongly suggest that when eigenchannel or NAP channel compensation are used, then feature mapping becomes unnecessary.

In the BUT systems, 8 feature mapping channels were found by unsupervised iterative re-clustering of conversations [15], primed with the TNO feature mapping labels (CDMA, GSM, carbon button, electret per gender), as used in SRE-2005. These were augmented with 6 channels determined from SRE-2004 labels (cellular, cordless, standard per gender). The TNO feature mapping used 16 classes, and was trained with balanced quantities from Switchboard (640 speakers) and Fisher (1000 speakers) databases.

### C. Eigenchannel GMM

We adopted the term 'eigenchannel' as used in speaker recognition from Kenny [6]. It was introduced to the NIST SRE by SDV in 2004 [16], revisited by Kenny [17], [18] and Vogt [19] in SRE 2005, and again by several sites in various forms in SRE-2006 [20].

In our Gaussian mixture model (GMM) system [14], speaker models were trained in the usual way by adapting from a universal background model (UBM [4]) by maximum a-posteriori (MAP) adaptation [21]. Only means of Gaussian components are adapted.

In the following, we will use the notion of *supervectors*[2]: Since our GMMs differ only in means, each model can be represented by the concatenation of all the mean vectors of all the Gaussians in the model. (We normalized each mean by the corresponding standard deviation.)

In eigenchannel adaptation, a model that has been trained under one channel condition, may be adapted towards a different channel condition of new test data, to reduce mismatch when the speaker is the same. Importantly, the adaptation must be constrained so that adaptation between different speakers is suppressed. This constraint is effected by adapting GMM models in supervector space, but only in a very small[3] subspace.

The adaptation is effected by maximizing (with a single iteration of the Expectation Maximization (EM) algorithm [21]) the MAP-criterion, $P(\{f_t\}|\mathbf{m}+\mathbf{V}\mathbf{x})P(\mathbf{x})$, w.r.t. the low-dimensional 'channel mismatch' vector $\mathbf{x}$ [16], [14]. Here, $\{f_t\}$ is the sequence of acoustic feature vectors in the test segment, $\mathbf{m}$ is the supervector representing the original model, $\mathbf{V}$ is a low-rank matrix that spans the adaptation subspace, and $P(\mathbf{x})$ is a zero-mean, unit-covariance Gaussian prior on the channel mismatch. In later experiments, we found the prior to be unimportant and that the MAP-criterion could be replaced by a simpler ML-criterion, by ignoring the prior. The adaptation subspace $\mathbf{V}$ was trained via the same eigen-analysis that was used to find the NAP-subspace, see Section II-F1.

In the variant of this system without T-norm (test normalization), the score for each trial was calculated as $\log P(\{f_t\}|\mathbf{m}_a) - \log P(\{f_t\}|\mathbf{U}_a)$, where $\mathbf{m}_a$ and $\mathbf{U}_a$ are the independently adapted target and universal background models. In the T-normed variant, the score was normalized in the usual way [22], but with each T-norm model also independently adapted. The EM-algorithm for adaptation of multiple T-norm models was streamlined by using the state occupancy probabilities of the UBM for all models, as proposed by [19].

The BUT eigenchannel GMM system and its interaction with various feature-space compensations such HLDA and feature mapping is analysed in more detail in [14].

### D. GMM-SVM

In this type of system, GMM supervectors, as described in the previous section, are extracted not only from target-model training speech segments, but also for all other background and test speech segments. In other words, each speech segment (conversation side) is represented by a single GMM

[2] Supervectors are just rather large vectors, where 'super' serves to distinguish them from the much smaller short-time feature vectors.

[3] In this case the subspace was 30-dimensional while the full supervector dimension was almost 80000.

supervector. The target and background supervectors are then used to train support vector machine (SVM) speaker models against which the test supervectors are scored [23], [24]. The SVM uses a linear kernel in supervector space. Each SVM is trained using the single available positive example from the target speaker, and many[4] negative examples from a pool of background speakers.

All our SVM sub-systems used NAP as a preprocessing step before SVM training. This is described in Section II-F.

### *E. MLLR-SVM*

This type of system makes use of large vocabulary continuous speech recognition (LVCSR). Previous work [25] has already shown that the adaptation matrices that LVCSR systems use to adapt towards new speakers are excellent features for speaker recognition.

The sub-systems in this paper used the coefficients from constrained maximum likelihood linear regression (CMLLR) and maximum likelihood linear regression (MLLR) transforms, as estimated by the LVCSR system developed in AMI project[5] submitted to NIST Rich Transcription 2005 evaluations [26]. This adaptation was 'supervised' by using the ASR transcripts[6], as made available by NIST for all speech data in SRE-2005 and 2006. Since NIST did not provide pronunciation dictionary, we used the AMI dictionary and we generated the missing pronunciations automatically. With this, we were able to generate the triphone alignment, to apply vocal tract length normalization (VTLN) and to estimate the coefficients of CMLLR and MLLR transformations.

These coefficients were normalized and concatenated into supervectors and then used with SVMs, exactly as described in the previous subsection for the GMM supervectors.

### *F. Nuisance attribute projection (NAP)*

All of our SVM sub-systems used NAP [7], [27] to remove unwanted channel or inter-session variability. There are different ways in which the NAP transform may be estimated and applied. We give here the general recipe that we applied in all of the STBU SVM systems. We also show how the eigenchannel adaptation matrix $\mathbf{V}$ was obtained.

*1) NAP training:* The data collection used in SRE-2004 was specifically designed to contain a large channel variability. Hence, as training material for the NAP-transforms we used whole conversation sides from the NIST SRE-2004. This data includes circa 310 speakers for most of which there are 10 or more conversation sides, or *sessions*. The steps for estimating the NAP transform are:

- Extract a supervector of dimension[7] $D_{\text{sv}}$ for each session of each speaker.
- For each speaker, calculate the mean supervector over all of the available supervectors of that speaker. Then subtract the mean from all of the vectors for that speaker. Pooling all these difference vectors then gives a large matrix $\mathbf{D}$ of supervectors from which most of the speaker variability has been removed, but where the inter-session (or nuisance) variability remains. The matrix $\mathbf{D}$ has dimensions $D_{\text{sv}} \times N_{\text{ses}}$, where $N_{\text{ses}}$ is the total number of sessions.
- Select the NAP transform dimension, denoted as $D_{\text{NAP}}$. We typically used $D_{\text{NAP}} = 40$, but this dimension should be chosen empirically as the one which gives best results.
- Now perform a principal component analysis (PCA) on $\mathbf{D}$. That is, we need to find the $D_{\text{NAP}}$ principal eigenvectors of the normalized scatter matrix[8] $\frac{1}{N_{\text{ses}}}\mathbf{D}\mathbf{D}^T$. Since the number of session vectors is typically several thousand, and the supervector dimension can be in the tens of thousands, some careful engineering may be needed to find these eigenvectors on machines of limited memory and CPU capacity. Some hints are given in Section II-H. We denote the $D_{\text{sv}} \times D_{\text{NAP}}$ matrix of principal eigenvectors as $\mathbf{E}$.
- Since an iterative eigenvector algorithm typically gives approximate solutions, it is a good precaution to normalize and mutually orthogonalize the columns of matrix $\mathbf{E}$, for example by *singular value decomposition* (SVD) of $\mathbf{E}$. If the eigenvectors are not orthonormal, the NAP-transform fails to project the nuisance subspace away completely.

*2) Eigenchannel matrix:* If the ML-version (without channel mismatch prior) of eigenchannel adaptation is used, it suffices to simply set $\mathbf{V} = \mathbf{E}$, where $\mathbf{V}$ is the matrix mentioned in Section II-C. However if MAP-adaptation is used, then each column $j$ of $\mathbf{V}$ should be scaled by $\sqrt{2e_j}$, where $e_j$ is the corresponding eigenvalue. (Directions in nuisance subspace with relatively smaller variances are thereby allowed to adapt to a lesser extent.)

*3) NAP-projection:* Once the orthonormal[9] NAP-subspace $\mathbf{E}$ has been trained as explained above, we may use it to train SVM speaker models that are more robust against inter-session variability. The basic NAP-transform is designed to be applied with linear-kernel SVMs. The transform must be applied to all supervectors (target and background) before they are used in SVM model training. That is, each supervector $\mathbf{v}$ is transformed as:

$$\mathbf{v}' = \mathbf{v} - \mathbf{E}(\mathbf{E}^T\mathbf{v}), \tag{1}$$

where $^T$ denotes transpose. By orthonormality, this transformation is idempotent [27]. This means it is not necessary to also NAP-transform the test supervectors[10], before they are scored against the SVM models. Finally, note that the NAP transform should be applied *before* SVM training. It does not

[4] Background size was of the order of 2000, which is much smaller than the supervector dimension. In practice this always results in SVM models with zero training errors. This makes selection of the SVM regularization constant irrelevant.

[5] See http://www.amiproject.org

[6] from a different English LVCSR system

[7] For GMM supervectors the dimension is the acoustic feature dimension times the number of GMM components. Numerical values are given in Table I.

[8] $\mathbf{D}$ has zero mean, so that this normalized scatter matrix acts as estimate of within-speaker covariance.

[9] $\mathbf{E}^T\mathbf{E} = \mathbf{I}$

[10] It would also not matter if this operation *was* repeated because of the idempotence.

help to apply the NAP-transform afterwards to test vectors or to models that have been trained on unprojected data.

### G. Division of training data

Although not all STBU sites had the same speech databases at their disposal, a general division of training data was made early in the design stage to which all sites adhered. Starting with the most recent collection, we used: *SRE-2005* exclusively for sub-system development testing, calibration and fusion; *SRE-2004* for eigenchannel, NAP, UBM, T-norm and rank normalization; *SRE 1999–2003, Fisher, Switchboard* for UBM training, feature mapping, SVM background, and T-norm.

### H. Some notes on computational efficiency

For experiments with these complex systems and large test databases it is important to have fairly efficient implementations of the various algorithms. Here we give some hints:

- Store the top-$N$ Gaussian index for each speech frame, where typically $N = 5$ [4]. Note that for obtaining this index for a frame $f_t$, only the distance to the Gaussian centers needs to be evaluated, and the exponentiation can be postponed or even omitted in the GMM-SVM case.
- For MAP adaptation of GMM means, only the top-$N$ Gaussian components need to be evaluated in the 'expectation-step,' making this typically a factor 100 faster [16]. Since this needs to be performed for each test segment (in the GMM-SVM case), this makes a big difference.
- In the estimation of the NAP projection, rather than calculating the principal $D_{\text{NAP}}$ eigenvectors of $\mathbf{D}\mathbf{D}^T$, calculate the principal eigenvectors of $\mathbf{D}^T\mathbf{D}$ (which is much smaller), and left-multiply these by $\mathbf{D}$ afterwards.
- Using ARPACK or Matlab's `eigs()`, explicit calculation of $\mathbf{D}^T\mathbf{D}$ is not necessary, but rather a function $f(\mathbf{x}) = \mathbf{D}^T\mathbf{D}\mathbf{x}$ can be provided. This function can be calculated without transposing large matrices using $f(\mathbf{x}) = \big((\mathbf{D}\mathbf{x})^T\mathbf{D}\big)^T$.
- For training SVM models (e.g., using `libSVM` [28]), pre-compute the Gram (kernel) matrix between all background speakers. Then for each new target/T-norm speaker, only one row and column needs to be replaced in the Gram matrix. This speeds up SVM training with orders of magnitude.
- For SVM scoring, SVM models can be folded, or compacted [5], into a single vector by calculating a weighted sum of the support vectors. Evaluation of a score is then just calculation of an inner product and T-normalization is just a matrix-vector multiplication.

## III. Sub-systems and their diversity

In the fusion of sub-systems, we found it advantageous to include in each fusion several very similar, but not identical, systems. Indeed, in post-evaluation experiments we found that leaving any of the sub-systems out caused significant deterioration in performance. These sub-systems were different because each was built by a different team, using different front-ends, different development databases and somewhat different flavours of the subspace channel compensation techniques. See Table I for a summary of the main characteristics of the various sub-systems.

Some remarks not captured in the table are the following. In an attempt to compensate for asymmetric system design, SDV provided two similar sub-systems: A *reverse* system swapped test and train speech segments for each trial, but was otherwise the same as the *forward* system. Because one speech segment is used for training the model and the other for obtaining a score, this swapping makes the system more symmetric. Experiments have shown that fusing these to sub-systems leads to better performance.

The acoustical features from BUT, as well as the MLLR transform data, were used by SUN. SUN provided two versions of the MLLR-SVM system, differing in the number of MLLR transforms.

CMLLR and MLLR transforms were trained for each speaker. At first, CMLLR was trained with two classes (speech + silence). On top of it, MLLR with two (SUN) or three (BUT, SUN) classes (the two speech classes were obtained by automatic clustering on the LVCSR training data + silence) was estimated. Using more classes caused missing data problems for some files, and was found not to lead to better performance. Both CMLLR and MLLR transform matrices were estimated as block-diagonal in 13-coefficient wide streams.

## IV. Fusion, calibration and decisions

The crux of the STBU design was to *fuse* multiple sub-systems into a single effective system. By fusion we mean the following: Let $x$ represent a speaker detection trial[11] and let this trial be processed in parallel by $N$ sub-systems, each of which produces a real-valued output *score*, where more positive scores favour the target hypothesis (same-speaker) and more negative the non-target hypothesis (different-speakers) . The score of the $i$th sub-system is denoted $s_i(x)$. These scores are fused using linear combination:

$$s_{\text{f}} = s(x, \mathbf{w}) = w_0 + \sum_{i=1}^{N} w_i s_i(x) \tag{2}$$

where $s_{\text{f}}$ is the fused output score and $\mathbf{w} = [w_0, w_1, \ldots, w_N]$ is a vector of real-valued weights. Perhaps counter intuitively, some of the weights may be negative.

### A. Logistic regression

The fusion weights were obtained by *logistic regression* [29] training on a database of supervised scores. We used all 1conv4w-1conv4w trials of the NIST SRE-2005 for this purpose. It is important to note that all development of the sub-systems did not make use of any 2005 data. If for example, 2005 data had been used to train NAP/eigenchannel, then the scores produced by these systems on the same data would have been over-optimistic and therefore not suitable for training fusion and calibration weights.

[11] Recall a trial consists of two speech segments.

TABLE I
SUMMARY OF ALL SUB-SYSTEMS COMPONENTS. LEGEND TO DATA SOURCE: SW: SWITCHBOARD, S$nn$: NIST SRE-'$nn$, F1: FISHER RELEASE 1. FRAME SELECTION METHODS (A)–(E) ARE EXPLAINED IN SECTION II-A.

| Site | SDV | BUT | | | SUN | | TNO |
|---|---|---|---|---|---|---|---|
| System | GMM-SVM | GMM | GMM-SVM | MLLR-SVM | GMM-SVM | MLLR-SVM | GMM-SVM |
| Features | 12 MFCC, $\Delta$ | 12 MFCC+$C_0$, $\Delta^3$ | | 12PLP+$C_0$,$\Delta^3$ | 12 MFCC+$C_0$, $\Delta^3$ | 12PLP+$C_0$,$\Delta^3$ | 12 PLP + $\log E$, $\Delta$ |
| HDLA dimension | | 39 | 39 | 39 | 39 | 39 | |
| Frame selection | (b),(d) | (a)–(c) | (a)–(c) | (e) | (a)–(c) | (e) | (a) |
| $N_f$ | 24 | 39 | 39 | 39 | 39 | 39 | 26 |
| UBM sources | S99–S03 | S04 | S04 | | S04 | | SW, S01–S03, F1 |
| $N_G$ | 512 | 2048 | 512 | | 512 | | 512 |
| Feature mapping channels | | 14 | 14 | | 14 | | 16 |
| Relevance factor | 16 | 19 | 19 | | 19 | | 16 |
| $D_{\text{sv}}$ | 12288 | 79872 | 19968 | 1638 | 19968 | 1092, 1638 | 13312 |
| $D_{\text{NAP}}$ | 40 | 30 | 40 | 15 | 40 | 15 | 40 |
| SVM Background speakers | $> 2000$ | | 2866 | 310 | 2606 | 310 | 1640 |
| source | S99–S03 | | F1, S02 | S04 | F1 | S04 | SW, S01–S03, F1 |
| T/Rank norm speakers | T: 310 | T:260 | T: 1080 | | T: 300 | T: 310 | T: 310 |
| | | | R: 2866 | R:310 | | | |
| source | S04 | S02 | F1, S02 | S04 | F1 | S04 | S04 |

The aim of logistic regression training is two-fold: First, it should improve *discriminative* ability, i.e., the DET-curve of the fused system should be better than the DET-curves of all the input systems. This is clearly demonstrated in Figs. 2, 3 and 6, which compare DET-plots of sub-systems against their fusion. Secondly it should *calibrate* the output score, so that it functions as a well-calibrated *log-likelihood-ratio*. That is, the training strives to achieve

$$s_{\text{f}} \approx \log \frac{P(s_{\text{f}}|H_{\text{tar}})}{P(s_{\text{f}}|H_{\text{non}})} \tag{3}$$

where $H_{\text{tar}}$ and $H_{\text{non}}$ denote target and non-target hypotheses respectively [30]. With a linear fusion such as (2), the degrees of freedom, which may be adjusted to optimize calibration, effectively form an affine transform—i.e., scores can be scaled and shifted. Scaling and shifting of scores does not affect discrimination and does not change the DET-plot.

There is a subtle difference between our use of logistic regression and the way in which it is traditionally applied in many other pattern recognition problems [31]. As mentioned, we train the fused score to function as a *log-likelihood-ratio*, while in other problems it is appropriate to train the score to function as *posterior log-odds*:

$$s'_{\text{f}} \approx \log \frac{P(H_{\text{tar}}|s'_{\text{f}})}{P(H_{\text{non}}|s'_{\text{f}})} = \log \frac{P_{\text{tar}}}{1-P_{\text{tar}}} + \log \frac{P(s'_{\text{f}}|H_{\text{tar}})}{P(s'_{\text{f}}|H_{\text{non}})} \tag{4}$$

In other words, the traditional posterior log-odds, $s'_{\text{f}}$ and our log-likelihood-ratio, $s_{\text{f}}$, differ essentially in an additive term, namely the *prior log-odds*,

$$\operatorname{logit} P_{\text{tar}} = \log \frac{P_{\text{tar}}}{1-P_{\text{tar}}} \tag{5}$$

As is shown below, this is easily handled by a small modification of the traditional logistic regression objective function. Let $\mathcal{X}_{\text{tar}}$ and $\mathcal{X}_{\text{non}}$ respectively represent sets of target and non-target trials. Our logistic regression objective function is:

$$\begin{aligned} \mathcal{O}(\mathbf{w}, P_{\text{tar}}) = {} & \frac{P_{\text{tar}}}{\|\mathcal{X}_{\text{tar}}\|} \sum_{x \in \mathcal{X}_{\text{tar}}} \log(1 + e^{-s(x,\mathbf{w}) - \operatorname{logit} P_{\text{tar}}}) \\ & + \frac{1-P_{\text{tar}}}{\|\mathcal{X}_{\text{non}}\|} \sum_{x \in \mathcal{X}_{\text{non}}} \log(1 + e^{s(x,\mathbf{w}) + \operatorname{logit} P_{\text{tar}}}) \end{aligned} \tag{6}$$

where $\|\mathcal{X}\|$ denotes the number of trials in set $\mathcal{X}$. Note that the objective is parameterized by the target prior $P_{\text{tar}}$. This adaptation of the logistic regression objective function allows one to set the parameter $P_{\text{tar}}$ independently of the proportion of target trials in the training database, to match the target prior of an envisaged application of the fusion. Since the purpose of this fusion was to optimize for the NIST SRE $C_{\text{det}}$ objective, we set [32]

$$\operatorname{logit} P_{\text{tar}} = \operatorname{logit} P'_{\text{tar}} + \log \frac{C_{\text{miss}}}{C_{\text{fa}}} \tag{7}$$

where $(P'_{\text{tar}}, C_{\text{miss}}, C_{\text{fa}}) = (0.01, 10, 1)$ are the parameters specified by the evaluation plan[12]. This gives $P_{\text{tar}} = 0.0917$. In experiments over a few different NIST SRE evaluation sets, we have found that, although performance of the logistic regression is relatively insensitive to the parameter $P_{\text{tar}}$, it does help to set it to the above value.

On the other hand, if the fusion is to be designed to optimize for the new $C_{\text{llr}}$ objective [32], [33], which was adopted as a secondary evaluation objective in the most recent NIST SRE Evaluation plan[13], then it would be better to choose $P_{\text{tar}} = 0.5$. Indeed, if (6) is reformulated as a function of the scores, rather than of $\mathbf{w}$, then at $P_{\text{tar}} = 0.5$, it is just the $C_{\text{llr}}$ objective.

At a fixed value of $P_{\text{tar}}$, the objective $\mathcal{O}(\mathbf{w}, P_{\text{tar}})$ is a convex function of $\mathbf{w}$, and it has a global minimum. This means it can be efficiently optimized with, for example, conjugate-gradient methods. We implemented a conjugate-gradient algorithm in Matlab, based on the work of Minka[14], but adapted to our variant of the objective function. This code is freely available as part of the FoCal toolkit[15].

### B. Missing trials

We had the complication that not all sub-systems were able to contribute a score for each trial, because of failure to detect speech in training or test segment, or lack of transcription. This necessitated a two step fusion strategy:

[12] See http://www.nist.gov/speech/tests/spk/

[13] See http://www.nist.gov/speech/tests/spk/2006/.

[14] See http://www.stat.cmu.edu/~minka/papers/logreg/

[15] See http://www.dsp.sun.ac.za/~nbrummer/focal/

1) First, each sub-system on its own was subjected to an affine calibration transformation[16], also trained via logistic regression, with $P_{\mathrm{tar}} = 0.5$. This calibration gave the scores a log-likelihood-ratio interpretation. The training data for this calibration were all trials that each sub-system could contribute out of the SRE-2005 (1conv4w-1conv4w) trials.
2) Next, scores (log-likelihood-ratios) of *zero* were inserted for all missing trials. Now, all sub-systems had valid scores for all trials and the fusion could be trained as explained above.

### C. Decisions

The beauty of a score that is calibrated so that approximation (3) holds is, that decisions with near-optimal expected cost can be made by using standard, theoretically determined score thresholds.

In past years, it was standard practice for NIST SRE participants to empirically determine score thresholds by optimizing average $C_{\mathrm{det}}$ performance over a database of supervised scores. This strategy indeed often worked well for the particular operating point defined via the $C_{\mathrm{det}}$ parameters. But if decisions at different operating points (different prior or costs) were required for applications other than the NIST SRE, then the threshold optimization procedure would have to be repeated.

The advantage of calibrated scores is that the empirical optimization, e.g. via logistic regression, over the supervised database needs to be performed once only. Thereafter, theoretical thresholds can be used to give good performance over a wide range of operating points [33]. If the goal is to make decisions that optimize $C_{\mathrm{det}}$, then the theoretical threshold is just the negative of (7):

$$\theta_{\mathrm{DET}} = -\operatorname{logit} P'_{\mathrm{tar}} - \log \frac{C_{\mathrm{miss}}}{C_{\mathrm{fa}}} \tag{8}$$

For $(P'_{\mathrm{tar}}, C_{\mathrm{miss}}, C_{\mathrm{fa}}) = (0.01, 10, 1)$, this gives $\theta_{\mathrm{DET}} = 2.29$. The decision rule is then:

$$\begin{aligned} s_{\mathrm{f}} &\geq \theta_{\mathrm{DET}} \mapsto \mathsf{accept}, \\ s_{\mathrm{f}} &\leq \theta_{\mathrm{DET}} \mapsto \mathsf{reject}. \end{aligned} \tag{9}$$

This new calibration-based strategy has indeed worked well, as demonstrated by small $C_{\mathrm{det}} - C_{\mathrm{det}}^{\mathrm{min}}$ discrepancies in the system submitted by SDV in the NIST SRE-2005, as well as for 5 of the best-performing systems[17] in the NIST SRE-2006, all of which used logistic regression-based calibration with a 2.29 threshold.

### D. Non-linear calibration (STBU-3)

As mentioned above, we are concerned with optimizing the discriminative ability (DET-curves), as well as the calibration, or *actual decision-making ability* of our scores. The traditional evaluation tools which are applied to analyse NIST SRE results include both (i) DET-curves to analyse discriminative ability over a *wide operating range* and (ii) $C_{\mathrm{det}}$ to analyse actual decision-making ability at a *fixed operating point*. The new $C_{\mathrm{llr}}$ metric serves to fill this gap: It evaluates average actual decision-making ability of log-likelihood-ratio scores, over a *wide operating range*. For a tutorial introduction to $C_{\mathrm{llr}}$ see [32] and for a reference implementation to calculate $C_{\mathrm{llr}}$ and $C_{\mathrm{llr}}^{\mathrm{min}}$, see the above-mentioned FoCal toolkit.

With our submissions STBU-1 and STBU-3, we tried to optimize calibration performance respectively for the traditional $C_{\mathrm{det}}$ and for the new $C_{\mathrm{llr}}$. STBU-1 was a straight-forward linear fusion (2) optimized with logistic regression with the parameter $P_{\mathrm{tar}} = 0.0917$. As explained, this fusion effects an affine calibration transformation.

STBU-3 took the score the output, $s_{\mathrm{f}}$, of STBU-1 and then subjected it to a further non-linear calibration stage. That is, the score of STBU-3 was obtained by:

$$s_{\mathrm{c}}(s_{\mathrm{f}}) = \log \frac{\alpha(e^{s_{\mathrm{f}}} - 1) + 1}{\beta(e^{s_{\mathrm{f}}} - 1) + 1} \tag{10}$$

where $0 < \beta < \alpha < 1$. This is a strictly increasing sigmoid function, which saturates below at approximately $-\operatorname{logit} \alpha$ and above at approximately $-\operatorname{logit} \beta$. The parameters $\alpha$ and $\beta$ are likewise found by optimizing the logistic regression objective, but here our aim was to optimize for $C_{\mathrm{llr}}$ rather than $C_{\mathrm{det}}$, so we set the parameter $P_{\mathrm{tar}} = 0.5$. Code for performing this optimization[18] is also available in the FoCal toolkit, as well as a derivation for the particular form of this saturating nonlinearity.[19] As shown in Table IV in the results section, the STBU-3 strategy did indeed improve calibration as measured by $C_{\mathrm{llr}}$.

## V. Results and discussion

### A. Comments on individual systems

In the development of individual systems, many configurations and parameters were tested and it is not possible to cover everything in this paper. We will therefore concentrate on the most important findings. The results will be presented on DET plots on 2006 data in Fig. 1:

1) Compare the influence that eigenchannel adaptation has on the GMM system (left) to the influence of NAP on the GMM-SVM (middle), as both techniques have similar underlying principles. We have found that, while eigenchannel greatly helps in the GMM system (and actually makes feature mapping unnecessary [14]), NAP helps in the GMM-SVM but to a much smaller extent. We attribute this to the fact that linear-kernel SVM models orient the score projection axis approximately perpendicular to the subspace spanned by all the background supervectors, which also includes much channel variation.
2) NAP in the MLLR-SVM sub-system (right) also helps, but it seems that SVM itself is able to exploit the

[16] This is the same as a fusion with a single input.

[17] NIST SRE rules prohibit publishing explicit performance details of other participants.

[18] Because of the saturation, the objective function may become non-convex. This makes it harder to optimize and it may fail to converge if not appropriately initialized.

[19] See http://www.dsp.sun.ac.za/~nbrummer/focal/cllr/calibration/s_cal/derivation.pdf

speaker-discriminative information in LVCSR adaptation matrices to some extent.

### *B. Fused systems and their results*

Three fused systems were submitted to the evaluation. The primary submission, STBU-1U (unsupervised adaptation mode) is an 11-fold fusion of:

1) GMM-SVM forward, T-normed (SDV)
2) GMM-SVM reverse, T-normed (SDV)
3) Eigen-channel GMM (BUT)
4) Eigen-channel GMM T-normed (BUT)
5) GMM-SVM T-normed (BUT)
6) MLLR3-SVM (BUT)
7) GMM-SVM T-normed (SUN)
8) MLLR2-SVM (SUN)
9) MLLR3-SVM (SUN)
10) GMM-SVM T-normed, without unsupervised adaptation (TNO)
11) GMM-SVM T-normed, with unsupervised adaptation (TNO)

For the non-adaptive variant STBU-1, indicated as STBU-1N in this paper, we simply omitted the last sub-system.

The second submission, STBU-2, is the same as STBU-1 in all respects, except that the eigenchannel GMM sub-systems were omitted. This makes this STBU-2 a pure fusion of SVM sub-systems. The third submission, STBU-3 is the same as STBU-1, except that the non-linear calibration described in Section IV-D was added.

Table II describes results on the primary condition (English only trials) for development data (SRE-2005) and for evaluation data (SRE-2006). Results are reported for all sub-systems, together with fused results which are with (U) and without (N) unsupervised adaptation. Fig. 2 presents the results graphically, where curves for GMM, GMM-SVM, and MLLR-SVM are grouped to keep the legend size manageable. Note how the curves for SRE-2006 are rotated clockwise w.r.t. the curves for SRE-2005. The little cusps in the MLLR-SVM curves are a side-effect of the zero-insertions discussed in Section IV-B.

Table III describes results on all trials from development and evaluation data. Only results of the best sub-system from each category is presented. Fig. 3 presents the results graphically with the same grouping of individual systems.

A comparison of the calibration performances of STBU-1 versus STBU-3 is given in table IV, as measured[20] on all 2006 1conv4w-1conv4w trials (without unsupervised adaptation). The *fixed-operating-point* calibration performance can be judged by the discrepancy between $C_{\text{det}}$ and $C_{\text{det}}^{\text{min}}$, indeed as planned, STBU-1 performed better than STBU-3. Conversely, the *general* calibration as judged by the discrepancy between $C_{\text{llr}}$ and $C_{\text{llr}}^{\text{min}}$ shows STBU-3, described in Section IV-D, to be better than STBU-1.

Although the calibration performance of the STBU system was good enough to make it competitive with the other submissions in the NIST SRE-2006, we note that the calibration performance in this evaluation was somewhat poorer for most participants as compared to the 2005 and 2004 evaluations. It is unlikely that this problem can be solved within the fusion and calibration paradigm presented here. Rather one may have to improve the sub-systems and make them more robust against changes in the nature of the speech data.

TABLE II
RESULTS OF THE SUB-SYSTEMS AND THE SUBMITTED ONE ON PRIMARY CONDITION: ENGLISH TRIALS.

| **system** | **SRE-2005 data** | | **SRE-2006 data** | | |
|---|---|---|---|---|---|
| | $C_{\text{det}}^{\text{min}}$ | EER | $C_{\text{det}}^{\text{min}}$ | EER | $C_{\text{det}}$ |
| GMM (BUT) | .0174 | 3.88% | .0178 | 3.44% | |
| GMM T-norm (BUT) | .0170 | 4.27% | .0159 | 3.44% | |
| GMM-SVM (SUN) | .0153 | 4.19% | .0171 | 3.61% | |
| GMM-SVM (BUT) | .0158 | 4.66% | .0185 | 3.71% | |
| GMM-SVM-U (TNO) | .0116 | 3.72% | .0185 | 3.81% | |
| GMM-SVM (TNO) | .0178 | 5.17% | .0190 | 4.10% | |
| GMM-SVM For (SDV) | .0221 | 6.05% | .0227 | 4.91% | |
| GMM-SVM Rev (SDV) | .0220 | 6.10% | .0238 | 5.18% | |
| MLLR3-SVM (SUN) | .0212 | 6.05% | .0218 | 4.49% | |
| MLLR3-SVM (BUT) | .0196 | 6.17% | .0220 | 4.78% | |
| MLLR2-SVM (SUN) | .0264 | 7.50% | .0270 | 5.56% | |
| STBU-1U | .0070 | 2.98% | .0132 | 2.26% | **0.0154** |
| STBU-1N | .0096 | 3.21% | .0126 | 2.32% | 0.0155 |
| STBU-2U | .0073 | 3.17% | .0147 | 3.07% | 0.0210 |
| STBU-2N | .0099 | 3.59% | .0147 | 3.07% | 0.0210 |
| STBU-3U | | | .0132 | 2.27% | 0.0161 |
| STBU-3N | | | .0126 | 2.32% | 0.0160 |

TABLE III
THE BEST PERFORMING SUB-SYSTEMS FROM EACH CATEGORY AND THE SUBMITTED RESULTS ON ALL TRIALS.

| **system** | **SRE-2005 data** | | **SRE-2006 data** | | |
|---|---|---|---|---|---|
| | $C_{\text{det}}^{\text{min}}$ | EER | $C_{\text{det}}^{\text{min}}$ | EER | $C_{\text{det}}$ |
| GMM (BUT) | .0201 | 4.83% | .0283 | 5.40% | |
| GMM-SVM (TNO) | .0192 | 5.77% | .0285 | 6.04% | |
| MLLR-SVM (BUT) | .0224 | 7.15% | .0327 | 7.57% | |
| STBU-1U | .0085 | 3.50% | .0208 | 3.30% | 0.0249 |
| STBU-1 | .0114 | 3.97% | .0214 | 3.83% | 0.0263 |

### *C. Unsupervised adaptation*

Unsupervised adaptation is an ‘operating mode’ of processing the NIST speaker recognition trials. In this mode, the available speech for a particular trial is extended with all earlier speech trials that include the same speaker model as the current trial. The trial index files are built such that (target) test segments are ordered by recording date for the same model speaker. The operating mode was proposed by Claude Barras [34] at the SRE-2003 workshop, adopted in the following NIST SRE plan as an optional mode, analysed separately. The rationale for this mode was that for certain applications, such as access authentication, there will typically be many target trials available which can provide the system with more speech of the target speaker so that better models can be formed [35].

For reasons which we will discuss below, successful application in a NIST SRE is hard [36], [37], but it finally succeeded in SRE-2005 [36]. Although, that year, only one participant had attempted to run the unsupervised adaptation mode, it still was considered an interesting research area, so it was decided that in SRE-2006, unsupervised adaptation mode

[20] Recall sub-systems were developed on 2004 and earlier data, fusion and calibration was trained on 2005 data, and this test was performed on new unseen 2006 data.

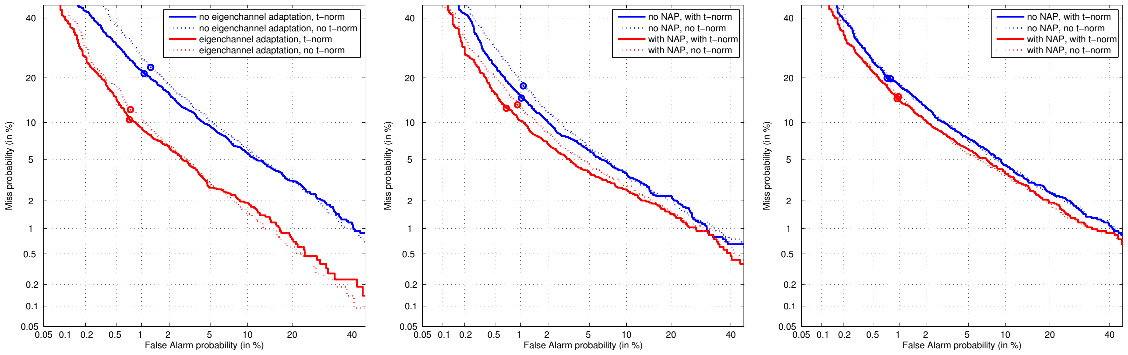


Fig. 1. Comparison of improvement with eigenchannel adaptation in the GMM system (left), NAP in GMM-SVM (middle) and NAP in MLLR-SVM (right). Results from SRE-2006, English-only trials. Circles indicate the $C_{\mathrm{det}}^{\mathrm{min}}$ operating point.

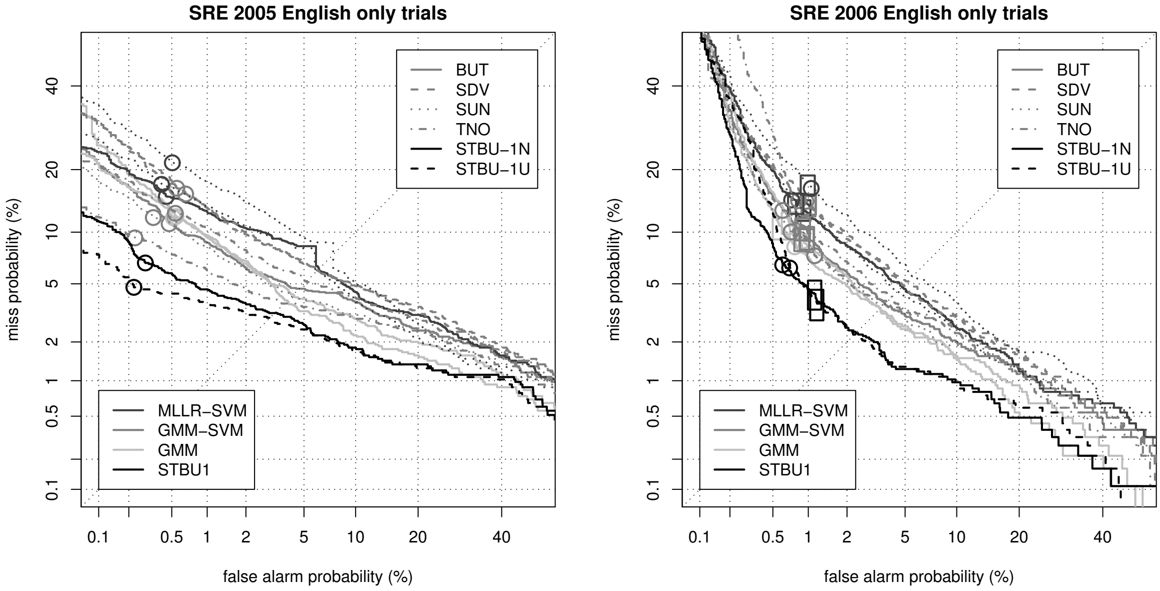


Fig. 2. DET curves for individual and merged systems, English only trials. MLLR-SVM, GMM-SVM and GMM sub-systems are grouped by colour. Left panel shows results on SRE-2005 data, where circles indicate $C_{\mathrm{det}}^{\mathrm{min}}$ operating point. Right panel shows SRE-2006 results, with additional boxes indicating 95 % confidence interval around the $C_{\mathrm{det}}$ operating point, based on calibration with SRE-2005 trials. The line type shows the site origin. STBU fusion results are in black, with a dashed curve for the unsupervised adaptation mode.

TABLE IV
COMPARISON OF CALIBRATION OF STBU-1 VS STBU-3, SRE-2006 ALL TRIALS.

| SYSTEM | $C_{\mathrm{llr}}$ | $C_{\mathrm{llr}}^{\mathrm{min}}$ | $C_{\mathrm{det}}$ | $C_{\mathrm{det}}^{\mathrm{min}}$ |
|---|---|---|---|---|
| STBU-1 | 0.198 | 0.152 | 0.0263 | 0.0214 |
| STBU-3 | 0.188 | 0.152 | 0.0274 | 0.0214 |

results could be entered as *primary system*.

There are different approaches to performing unsupervised adaptation, ranging from simple threshold-based inclusion of the test segment as extra training to score-weighted adaptation of the current model [35], [34], [37], [38], but all of them depend on proper calibration of the scores. This means that the *calibration will influence the position and shape* of the DET curve, as well as $C_{\mathrm{det}}$ and $C_{\mathrm{llr}}$. Further, as has been pointed out earlier [34], [36], the *evaluation priors* of target and non-target trials, as well as the number of target-trials for each model speaker determine the potential success of application of unsupervised adaptation. This is different from the 'normal mode' of operation, where the evaluation priors do not determine the performance measures such as $C_{\mathrm{det}}$ and EER. A last major difference between the two operating modes is the influence of 'pathological data' in the evaluation. In the much appreciated data collection efforts and quality control it is inevitable, given the large amount of trials in evaluations (over 50 000 in SRE-2006), that there are speech files which contain little or no speech, are duplicates, or have the wrong language or speaker ID associated with it. For the 'normal mode' of operation this causes little problems, because in a standard post evaluation quality control procedure by NIST,

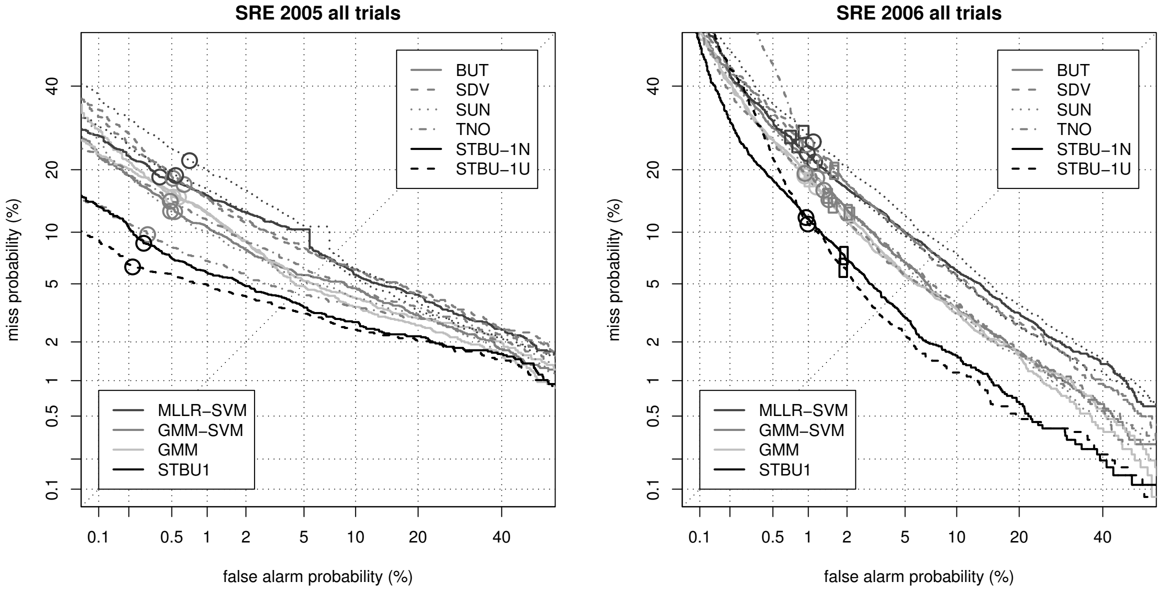


Fig. 3. DET curves for individual and merged systems, all trials. Colours, symbols and line type are the same as for Fig. 2.

trials involving these pathological files are discarded from further analysis. However, for the unsupervised adaptation mode, these pathological speech files can cause a major problem because the adaptive speaker model may deteriorate if such a file is not properly detected.

One sub-system (TNO) applied a simple adaptation scheme. It is based on earlier work [36] and extended to include the GMM-SVM-NAP technology. Basically, for each trial, the T-normed score $s$ is calculated. If $s$ exceeds a predetermined threshold $a$, the speech data in the test segment is used to MAP adapt the means in the GMM for the current model speaker, using a relevance factor $r$. The new means are used to build a new SVM, which is used for subsequent trials. The results for the development test (SRE-2005) and evaluation are summarized in Table V, and the DET-curves are shown in Fig. 4. Note, that these are the results of only one sub-system of the STBU submission. Qualitatively, the adaptation results are similar for the total system, but the effects are less pronounced due to the importance of several other sub-systems.

We tuned the parameters $a = 4$ and $r = 36$ to obtain optimum $C_{\rm det}^{\rm min}$ for SRE-2005, and applied these to SRE-2006. A speech file was classified as ‘potentially pathological’ if either the range of frame energy did not exceed 30 dB (assuming the file contains no speech) or if the SVM score, before T-norming, exceeded 0.95 (an assumed copy of a speech segment). For these trials, no adaptation was carried out. As it turns out, none of these trials survived the post evaluation quality control of NIST.

As can be observed from the table and the DET-curves, the discrimination performance increased dramatically for the development test (34 % relative drop in $C_{\rm det}$), but hardly at all for the evaluation (6 % relative drop in $C_{\rm det}$). The ‘knee’ close to the decision operating point for SRE-2006 is typical of runs where adaptation has been applied too aggressively (low $a$ and $r$). It shows the effect of ‘false adaptations’ which spoil a speaker model and lead to over-optimistic scores for subsequent non-target trials. The 53966 trials in SRE-2006 lead to 5003 adaptations, of which 61.4 % were correct, 13.9 % false adaptations, and 24.7 % unknown, because these trials were later removed from the official scoring by NIST due to the various problems described earlier. Even though ‘only’ 2518 trials were removed from the original trial index file, 1223 of these (49 %) were used for adaptation of speaker models. On the other hand, of a potential 3612 target trials, only 14.9 % were missed for adaptation (see Table V).

As a post-evaluation experiment, ‘post1,’ we ran our adaptive mode parameters on the list of trials that were kept after the post evaluation quality control. Oddly enough, we observe from Table V that the performance *decreases* under this condition. Apparently, the ‘pathological files’ that plagued so many researchers during the evaluation, helped our sub-system in unsupervised adaptation mode. Perhaps some speakers who had enrolled twice under a different identity in the data collection process, and whose ‘non-target trials’ were later removed, actually helped in adaptation mode.

We attribute the poor adaptation performance to the high probability of False Adaptation [34], which is an order of magnitude larger than in the development test. This is not only due to miscalibration, but also because the DET-curve has a steeper slope. Indeed, optimizing the threshold as a post-evaluation experiment ‘post2’ to $a = 5$ leads to the expected larger benefit of unsupervised adaptation (21.5 % drop in $C_{\rm det}$), with a much lower False Adaptation probability.

### D. Language dependence

We have observed that the performance in the primary condition (English only trials, Table II), is much better than

TABLE V

PERFORMANCE MEASURES FOR THE TNO SUB-SYSTEM IN NORMAL AN UNSUPERVISED ADAPTATION MODES, FOR DEVELOPMENT TEST (NO CALIBRATION, FIXED THRESHOLD OF 3), EVALUATION AND POST-EVALUATION EXPERIMENT. ALL (POST QUALITY CONTROL) TRIALS ARE INCLUDED. TWO POST-EVALUATION EXPERIMENTS ARE INCLUDED AS WELL. THE LAST TWO COLUMNS INDICATE THE PROBABILITY OF FALSE ADAPTATION AND MISSED ADAPTATION, RESPECTIVELY.

| Mode | dataset | $C_{\rm det}$ | $C_{\rm det}^{\rm min}$ | EER | $C_{\rm llr}$ | $C_{\rm llr}^{\rm min}$ | $P_{\rm FalseAd.}$ | $P_{\rm missAd.}$ |
|---|---|---|---|---|---|---|---|---|
| Normal | SRE-2006 | 0.0335 | 0.0286 | 6.04 % | 0.262 | 0.220 | | |
| Adapt. | SRE-2006 | 0.0315 | 0.0290 | 5.48 % | 0.264 | 0.219 | 13.9 % | 14.9 % |
| Normal | SRE-2005 | 0.0198 | 0.0189 | 5.79 % | 0.629 | 0.220 | | |
| Adapt. | SRE-2005 | 0.0130 | 0.0124 | 4.38 % | 0.572 | 0.171 | 1.1 % | 13.8 % |
| Adapt. post1 | SRE-2006 | 0.0349 | 0.0316 | 6.06 % | 0.284 | 0.236 | 17.2 % | 20.0 % |
| Adapt. post2 | SRE-2006 | 0.0262 | 0.0227 | 4.73 % | 0.220 | 0.182 | 5.5 % | 25.4 % |

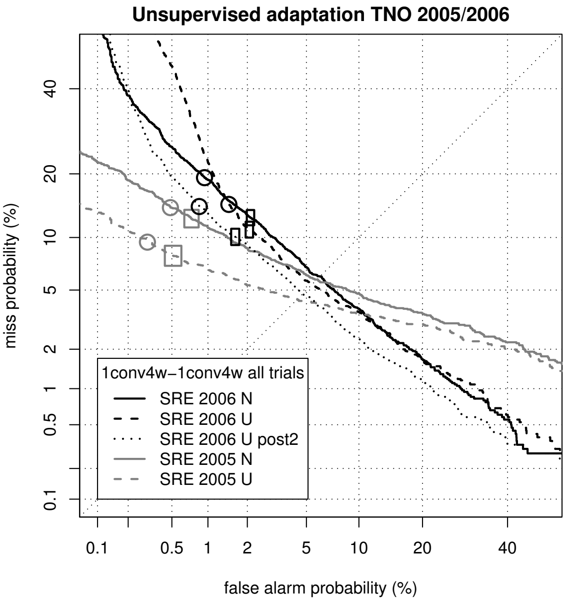


Fig. 4. DET-curves for the TNO sub-system in normal (solid lines) and unsupervised adaptation (dashed lines) modes, for evaluation and development test. Also included is a post-evaluation run with a more optimal threshold value (post2).

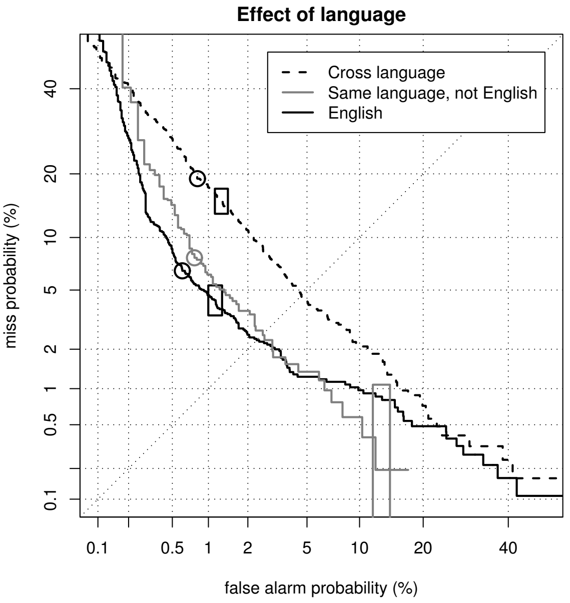


Fig. 5. DET-plots of the three different language condition analyzed in Table VI. The rectangle indicates the 95 % confidence interval around the decision point.

that of the entire evaluation (all trials, Table III). In this section we will analyse some language effects. A language dependence may be introduced by several parts of the system: the UBM, channel compensation, SVM background, score normalization and calibration. We split all valid trials of SRE-2006 into three conditions: *Same language English*, *Same language non-English* and *Cross language*. Note that by design of the evaluation, all cross-language trials involve English as one of the two spoken languages. In Table VI we summarize the important statistics of the three conditions.

Despite the low number of trials available for the non-English same-language condition, we can observe the following. The *discrimination* potential of the system seems similar for English and non-English same-language conditions, judged from a very similar EER, $C_{\rm det}^{\rm min}$ and $C_{\rm llr}^{\rm min}$. But the *calibration* for non-English trials is very poor ($C_{\rm det}$, $C_{\rm llr}$), compared to the English trials. This result suggests that the UBM and channel compensation components are less language dependent, but that there is a possible language dependence in score normalization and definitely in the calibration. Most sub-systems applied T-norm score normalization [39]. Because we applied predominantly English T-norm model speakers, we can imagine that non-English test segments will have lower scores for the T-norm models than the English test segments. This would lead to higher T-normed scores for non-English trials, for both target and non-target, such that the calibration is skewed towards more false alarms. Indeed, this is what is observed in Fig. 5.

A genuine discrimination loss is observed in the *cross language* trials. Interestingly, the calibration of the cross-language condition seems to be reasonable. This may be due to the fact that all cross-language target trials had English as one of the two speech segment languages. Apparently, having at least one English speech segment helps the calibration a lot.

All the described effects are qualitatively the same as observed for just a single sub-system (TNO) of the STBU fusion.

TABLE VI
LANGUAGE DEPENDENCE OF THE STBU-1 SYSTEM, FOR ENGLISH SAME-LANGUAGE TRIALS, NON-ENGLISH SAME LANGUAGE TRIALS AND CROSS LANGUAGE TRIALS.

| Language | $C_{\text{det}}$ | $C_{\text{det}}^{\text{min}}$ | EER | $C_{\text{llr}}$ | $C_{\text{llr}}^{\text{min}}$ | $N_{\text{tar}}$ | $N_{\text{non}}$ |
|---|---|---|---|---|---|---|---|
| English | 0.0155 | 0.0126 | 2.32 % | 0.148 | 0.101 | 1854 | 22159 |
| Non-English | 0.128 | 0.0154 | 2.54 % | 0.721 | 0.099 | 516 | 2857 |
| Cross language | 0.0277 | 0.0272 | 4,60 % | 0.199 | 0.180 | 1242 | 22820 |

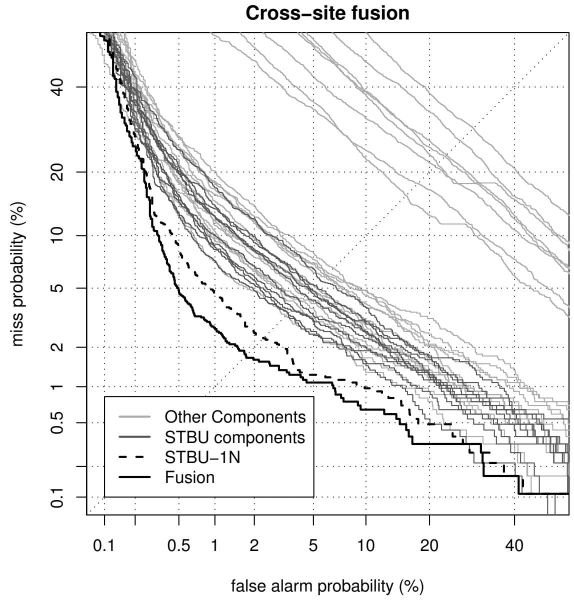


Fig. 6. Cross-site fusion: DET curves for individual and fused systems, for the English only trials condition of NIST SRE-2006.

## VI. CROSS-SITE FUSION

As a final demonstration of the power of fusing diverse subsystems, we increased the diversity and tripled the number of sub-systems by also including sub-systems, that performed well, from 6 other participating SRE-2006 sites. Together with 10 of the STBU sub-systems, this gave a total of 31 sub-systems, all non-adaptive. The fusion was trained on the supervised scores of all 1conv4w-1conv4w trials of SRE-2005 and then tested on the English 1conv4w-1conv4w trials of SRE-2006. See the DET-curves of Fig. 6, which shows (i) all 31 sub-systems, (ii) the original STBU-1 fusion, and (iii) the total fusion of all 31 systems. It is clear that the two fusions outperform any individual system, and that the bigger fusion (EER $= 1.7\,\%$) outperforms the original STBU fusion (EER $= 2.3\,\%$).

## VII. CONCLUSION

The STBU system has demonstrated a few important principles that were exploited in reaching state-of-the-art speaker detection performance. (i) GMMs and SVMs are still important basic workhorses in speaker recognition, but alternative strategies like MLLR-SVM are not to be ignored. (ii) An abundance of suitable development data is perhaps the most important resource. Without the SRE-2004 and SRE-2005 databases, developing, testing and calibrating the powerful subspace channel compensation would not have been possible. Until recently, speaker recognition had been all about training individual speaker models. The emphasis has now shifted to the data-driven training of methods that can discriminate between speakers—we are no longer just training speaker models in isolation, each on a few minutes of speech. We are now training whole systems on the hundreds of hours of speech in whole NIST SRE databases. This is exemplified not only by eigenchannel and NAP, but also by fusion, which likewise needs to be trained on entire SRE databases. (iii) Calibration, in order to make *actual* decisions, has always been important in the NIST evaluations, but this had previously been measured only at the same fixed $C_{\text{det}}$ operating point. The introduction of $C_{\text{llr}}$ has now widened the scope of the calibration challenge, and so far not only the STBU system, but several other SRE-2006 participants have met this challenge successfully.

Despite these successes, several problem areas remain. As our investigation into the influence of the spoken language in detection performance shows, there is a strong effect on our system's calibration if trials are not English, and there is a reduction in discrimination if the segments of the trials are spoken in different languages. Perhaps these issues can be resolved with techniques similar to the channel compensation approaches. The unsupervised adaptation mode of processing trials did not deliver the large benefit we had expected, and we attribute this to a calibration mismatch, to which adaptation is very sensitive, and to the mysterious clockwise rotation of the DET curve observed for all systems that perform well. There remains the unsolved question of why new data collections and acoustic conditions seem to have an effect of rotation of the DET-curve—maybe to a more 'natural' state of equal width target and non-target score distributions. Continuing data collections, evaluations and research may on the long term provide us with an answer.

## VIII. ACKNOWLEDGEMENTS

We would like to thank all SRE-2006 participants who provided their system results for Section VI, and granted permission to publish them anonymously. This work was partly supported by by European projects AMIDA (IST-033812) and Caretaker (FP6-027231), by Grant Agency of Czech Republic under projects No. 102/05/0278 and GP102/06/383, and by Czech Ministry of Education under project No. MSM0021630528. It is further partly supported by the project MultimediaN (http://www.multimedian.nl). MultimediaN is sponsored by the Dutch government under contract BSIK 03031.

# References

[1] P. Matějka, L. Burget, P. Schwarz, O. Glembek, M. Karafiát, J. Černocký, D. A. van Leeuwen, N. Brümmer, A. Strasheim, and F. Grézl, "STBU system for the NIST 2006 speaker recognition evaluation," in *Proc. ICASSP*, 2007, accepted for publication.

[2] G. R. Doddington, M. A. Przybocki, A. F. Martin, and D. A. Reynolds, "The NIST speaker recognition evaluation—Overview, methodology, systems, results, perspective," *Speech Communication*, vol. 31, pp. 225–254, 2000.

[3] D. A. van Leeuwen, A. F. Martin, M. A. Przybocki, and J. S. Bouten, "NIST and TNO-NFI evaluations of automatic speaker recognition," *Computer Speech and Language*, vol. 20, pp. 128–158, 2006.

[4] D. Reynolds, T. Quatieri, and R. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, pp. 19–41, 2000.

[5] W. M. Campbell, "Generalized linear discriminant sequence kernels for speaker recognition," in *Proc. ICASSP*, 2002, pp. 161–164.

[6] P. Kenny and P. Dumouchel, "Disentangling speaker and channel effects in speaker verification," in *Proc. ICASSP*, 2004, pp. 37–40.

[7] A. Solomonoff, W. Campbell, and I. BoardmanCampbell, "Advances in channel compensation for SVM speaker recognition," in *Proc. ICASSP*, vol. I, Philadelphia, PA, USA, Mar. 2005, pp. 629–632.

[8] M. A. Przybocki, A. F. Martin, and A. N. Le, "NIST speaker recognition evaluation chronicles—part 2," in *Proc. Odyssey 2006 Speaker and Language recognition workshop*, San Juan, June 2006.

[9] H. Hermansky and N. Morgan, "Rasta processing of speech," *IEEE Transactions on Speech and Audio Processing, special issue on Robust Speech Recognition*, vol. 2, no. 4, pp. 578–589, 1994.

[10] J. Pelecanos and S. Sridharan, "Feature warping for robust speaker verification," in *Proc. Speaker Odyssey*. Crete, Greece, 2001.

[11] N. Kumar, "Investigation of silicon-auditory models and generalization of linear discriminant analysis for improved speech recognition," Ph.D. dissertation, John Hopkins University, Baltimore, 1997.

[12] L. Burget, "Complementarity of speech recognition systems and system combination," Ph.D. dissertation, Brno University of Technology, Czech Republic, 2004.

[13] D. A. Reynolds, "Channel robust speaker verification via feature mapping," in *Proc. ICASSP*, 2003, pp. 53–56.

[14] L. Burget, P. Matějka, O. Glembek, P. Schwarz, and J. H. Černocký, "Analysis of feature extraction and channel compensation in GMM speaker recognition system," *IEEE Trans. on Audio, Speech and Language Processing*, 2007, accepted.

[15] M. Mason, R. Vogt, B. Baker, and S. Sridharan, "Data-driven clustering for blind feature mapping in speaker verification," in *Proc. Eurospeech*, Lisbon, Portugal, Sep. 2005, pp. 3109–3112.

[16] N. Brümmer, "Spescom DataVoice NIST 2004 system description," in *Proc. NIST Speaker Recognition Evaluation 2004*, Toledo, Spain, Jun. 2004.

[17] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Speaker and session variability in GMM-based speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 4, May 2007.

[18] ——, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 4, May 2007.

[19] R. Vogt, B. Baker, and S. Sridharan, "Modelling session variability in text-independent speaker verification," in *Proc. Interspeech*, 2005, pp. 3117–3120.

[20] R. Vogt and S. Sridharan, "Experiments in session variability modelling for speaker verification," in *Proc. ICASSP*, vol. 1, Toulouse, France, May 2006, pp. 897–900.

[21] J.-L. Gauvain and C.-H. Lee, "Maximum a posteriori esitimation for multivariate gaussian mixture observations of markov chains," *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 291–298, 1994.

[22] R. Auckenthaler, M. Carey, and H. Lloyd-Tomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Processing*, vol. 10, pp. 42–54, 2000.

[23] W. Campbell, D. Sturim, and D. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[24] W. Campbell, "A SVM/HMM system for speaker recognition," in *Proc. ICASSP*, Hong Kong, Apr. 2003, pp. 156–159.

[25] A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, "MLLR transforms as features in speaker recognition," in *Proc. Eurospeech*, Lisbon, Portugal, Sep. 2005, pp. 2425–2428.

[26] T. Hain, L. Burget, J. Dines, G. Garau, M. Karafiat, M. Lincoln, I. McCowan, D. Moore, V. Wan, R. Ordelman, and S. Renals, "The 2005 AMI system for the transcription of speech in meetings," in *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2006, vol. 3869, pp. 450–462.

[27] W. Campbell, D. Sturim, D. Reynolds, and A. Solomonoff, "Svm based speaker verification using a gmm supervector kernel and nap variability compensation," in *Proc. ICASSP*. Toulouse: IEEE, 2006, pp. 97–100.

[28] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," http://www.csie.ntu.edu.tw/∼cjlin/libsvm, 2001.

[29] S. Pigeon, P. Druyts, , and P. Verlinde, "Applying logistic regression to the fusion of the nist'99 1-speaker submissions," *Digital Signal Processing*, 2000.

[30] N. Brümmer, "Measuring, refining and calibrating speaker and language information extracted from speech," Ph.D. dissertation, Stellenbosch University, 2007.

[31] D. W. Hosner and S. Lemeshow, *Applied Logistic Regression*. John Wiley & Sons, 1989.

[32] D. A. van Leeuwen and N. Brümmer, "An introduction to application-independent evaluation of speaker recognition systems," in *Speaker Classification*, ser. Lecture Notes in Computer Science / Artificial Intelligence, C. Müller, Ed. Heidelberg - New York - Berlin: Springer, 2007, vol. 4343.

[33] N. Brümmer and J. du Preez, "Application-independent evaluation of speaker detection," *Computer Speech and Language*, vol. 20, pp. 230–275, 2006.

[34] C. Barras, S. Meigner, and J. L. Gauvain, "Unsupervised online adaptation for a speaker verification system over the telephone," in *Proc. Speaker Odyssey*, 2004.

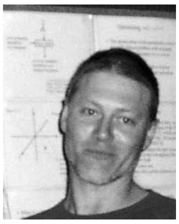
[35] N. Mirghafori and L. Heck, "An adaptive speaker verification system with speaker dependent a proiri deciasion thresholds," in *Proc. ICSLP*, 2002, pp. 589–592.

[36] D. A. van Leeuwen, "Speaker adaptation in the NIST speaker recognition evaluation 2004." in *Proc. Eurospeech*, 2005, pp. 1981–1984.

[37] E. G. Hansen, R. E. Slyh, and T. R. Anderson, "Supervised and unsupervised speaker adaptation in the nist 2005 speaker recognition evaluation," in *Proc. Odyssey 2006 Speaker and Language Recognition Workshop*, 2006.

[38] S.-C. Yin, P. Kenny, and R. Rose, "Speaker adaptation for factor analysis based speaker verification," in *Proc. Odyssey 2006 Speaker and Language recognition workshop*, San Juan, June 2006.

[39] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Processing*, vol. 10, pp. 42–54, 2000.



**Niko Brümmer** (M.Eng, University of Stellenbosch, 1988) is planning to submit his thesis entitled "Measuring, refining and calibrating speaker and language information extracted from speech," for a Ph.D, University of Stellenbosch in 2007. He has been employed as research engineer by Spescom DataVoice in South Africa, from 1990 to the present, on behalf of whom he has participated in 5 NIST Speaker Recognition Evaluations between 2000 and 2006, and also the NIST Language Recognition Evaluation 2005. His research interests include speaker and language recognition and the evaluation and improvement of pattern-recognition and machine-learning technologies via information theory.

**Lukáš Burget** (Ing. [MS]. Brno University of Technology, 1999, Ph.D. Brno University of Technology, 2004) is employed as assistant professor at at Faculty of Information Technology, University of Technology, Brno, Czech Republic. The topic of his PhD dissertation, that he successfully defended in November 2004, was: "Complementarity of Speech Recognition Systems an System combination". From 2000 to 2002, he was a visiting researcher at OGI Portland, USA under supervision of Prof. Hynek Hermansky. He is member of IEEE and ISCA. His scientific interests are in the field of speech processing, namely acoustic modeling for speech recognition.

**Jan "Honza" Černocký** (Ing. [MS] 1993 Brno University of Technology (BUT); Dr. [PhD] 1998 Université Paris XI and BUT) was with the Institute of Radio-electronics, BUT (Faculty of Electrical Engineering and Computer Science) as assistant professor from 1997. Since February 2002, he is with the Faculty of Information Technology (FIT), BUT as Associate Professor (Doc.) and Deputy Head of the Institute of Computer Graphics and Multimedia. With Prof. Hynek Hermansky he is leading the Speech@FIT group at FIT VUT. He supervises several PhD students, and coordinates Speech@FIT activities in several European and national projects. His research interests include signal processing, speech processing (very low bit rate coding, verification, recognition), segmental methods, data-driven determination of speech units and speech corpora. He is a member of IEEE and ISCA and serves on the board of Czechoslovak section of IEEE.

**Ondřej Glembek** (Ing. [MS]. Brno University of Technology, 2005) was student at Brno University of Technology, faculty of Electrical Engineering and Computer, later Faculty of Information Technology from 1999. From September till December 2003, he was at University of Joensuu, Finland as a participant of the Socrates/Erasmus program. From October till November 2004, he was working on a project concerning wavelet transforms at Izhevsk State Technical University, Izhevsk, Russia. From 2005, he is PhD student in Speech@FIT - he is concentrating on acoustic modeling for speech recognition, recognition of Czech and STK toolkit development.

**František Grézl** (Ing. [MS]. Brno University of Technology, 2000) is post-gradual student of Speech processing group at the Faculty of Information Technology (FIT), BUT since September 2000 and is planning to submit his doctoral thesis "Acoustic modeling for speech recognition" in summer 2007. He has been with the Anthropic speech processing group of Oregon Graduate Institute of Science and Technology, USA, with speech processing group at IDIAP research institute, Martigny, Switzerland and with ICSI International Computer Science Institute Berkeley, California under the AMI training programme. His main research interests include robust speech recognition and feature extraction.

**Martin Karafiát** (Ing. [MS]. Brno University of Technology, 2001) is post-gradual student in Speech@FIT at the Faculty of Information Technology (FIT), BUT since September 2001 and is planning to submit his doctoral thesis in autumn 2007. He was twice in the internship at University of Sheffield with Speech and Hearing Group, UK. Main reason for both internships was work on Large Vocabulary Continuous Speech Recognizers for two EU-projects M4 (Multimodal Meeting manager) and AMI (Augmented multiparty interaction). His research interest is speech recognition—especially speech recognition with large vocabulary, including feature transforms and novel feature extractions such as TRAPs.

**David A. van Leeuwen** (Ir. [MS] 1984 Delft University of Technology, Dr. [PhD] 1993 University of Leiden) is with TNO Human Factors since 1994. He has been active in the field of large vocabulary continuous speech recognition (evaluation, development of Dutch system), word spotting, and speaker and language recognition. He has organized several benchmark evaluations (LVCSR Fr/Ge/BrEng: EU SQALE in 1995, Forensic Speaker Recognition NFI-TNO in 2003, LVCSR Dutch: N-Best in 2008). He has participated in NIST SRE, LRE and RT evaluations since 2003. He has been a representative in several NATO IST Research task groups on speech technology since 2002, and an ISCA member since 1995.

**Pavel Matějka** (Ing. [MS]. Brno University of Technology, 2001) is PhD student at Institute of Radioelectronics, Faculty of Electrical Engineering and Communication and Department of Computer Graphics and Multimedia, FIT, BUT. He is planning to submit his doctoral thesis "Language identification based on phonetic cues" in summer 2007. He has been with the Anthropic speech processing group at Oregon Graduate Institute of Science and Technology, USA. He is member of IEEE and ISCA. His research interests include speaker recognition, language identification, speech recognition, namely phoneme recognition based on novel feature extractions (temporal patterns) and neural networks. He has been active also in keyword-spotting and on-line implementation of speech processing algorithms. He was finalist in Student paper contest at ICASSP2006 in Toulouse.

**Petr Schwarz** (Ing. [MS]. Brno University of Technology, 2001) is post-gradual student of Speech processing group at the Faculty of Information Technology (FIT), BUT since September 2001 and is planning to submit his doctoral thesis "Robust phoneme recognition" in 2007. He has been with the Anthropic speech processing group of Oregon Graduate Institute of Science and Technology, USA. He is member of IEEE and ISCA. His research interests include speech recognition, namely phoneme recognition based on novel feature extractions (temporal patterns) and neural networks. He has been active also in keyword-spotting and on-line implementation of speech processing algorithms.

**Albert Strasheim** (B.Sc, University of Stellenbosch, 2003, B.Eng, University of Stellenbosch, 2005) is a Masters student at the University of Stellenbosch, Digital Signal Processing Lab, supervised by Prof. Johan du Preez. His research interests include software engineering and parallel and distributed systems, specifically their application to pattern recognition and machine learning problems.

# COMPARISON OF SCORING METHODS USED IN SPEAKER RECOGNITION WITH JOINT FACTOR ANALYSIS

*Ondřej Glembek*[1], *Lukáš Burget*[1], *Najim Dehak*[2,3], *Niko Brümmer*[4], *Patrick Kenny*[2]

[1]Speech@FIT group, Faculty of Information Technology, Brno University of Technology, Czech Republic
[2]Centre de Recherche Informatique de Montréal (CRIM), Montréal, Canada
[3]École de Technologie Supérieure (ETS), Montréal, Canada
[4]Agnitio, Stellenbosch, South Africa
{glembek,burget}@fit.vutbr.cz, {najim.dehak,patrick.kenny}@crim.ca,
nbrummer@agnitio.es

## ABSTRACT

The aim of this paper is to compare different log-likelihood scoring methods, that different sites used in the latest state-of-the-art Joint Factor Analysis (JFA) Speaker Recognition systems. The algorithms use various assumptions and have been derived from various approximations of the objective functions of JFA. We compare the techniques in terms of speed and performance. We show, that approximations of the true log-likelihood ratio (LLR) may lead to significant speedup without any loss in performance.

***Index Terms***— GMM, fast scoring, speaker recognition, joint factor analysis

## 1. INTRODUCTION

Joint Factor Analysis (JFA) has become the state-of-the-art technique in the problem of speaker recognition[1]. It has been proposed to model the speaker and session variabilities in the parameter space of the Gaussian Mixture Model (GMM) [1]. The variabilities are determined by subspaces in the parameter space, commonly called the *hyper-parameters*.

Many sites used JFA in the latest NIST evaluations, however they report their results using different scoring methods ([2], [3], [4]). The aim of this paper is to compare these techniques in terms of speed and performance.

The theory about JFA and each technique is given in Sec. 2. Starting with the conventional frame-by-frame GMM evaluation in Sec. 2.1, where the whole feature file of each utterance is processed, the sections 2.2 to 2.5 describe methods which work with the collected statistics only and which differ mostly in the way they treat channel compensation. In Sec. 2.2, integration over the whole distribution of channel factors for the given test utterance is performed. In Sec. 2.3, the likelihood of each utterance given testing model is computed using a channel point estimate. In Sec. 2.4, the channel factor point estimate is estimated using UBM only. In Sec 2.5, the formula is further simplified by using the first order Taylor series approximation.

## 2. THEORETICAL BACKGROUND

Joint factor analysis is a model used to treat the problem of speaker and session variability in GMMs. In this model, each speaker is represented by the means, covariance, and weights of a mixture of $C$ multivariate Gaussian densities defined in some continuous feature space of dimension $F$. The GMM for a target speaker is obtained by adapting the Universal Background Model (UBM) mean parameters. In Joint Factor Analysis [2], the basic assumption is that a speaker- and channel- dependent supervector of means $\mathbf{M}$ can be decomposed into a sum of two supervectors: a speaker supervector $\mathbf{s}$ and a channel supervector $\mathbf{c}$

$$\mathbf{M} = \mathbf{s} + \mathbf{c}, \tag{1}$$

where $\mathbf{s}$ and $\mathbf{c}$ are normally distributed. In [5], Kenny et al. described how the speaker dependent supervector and channel dependent supervector can be represented in low dimensional spaces. The first term in the right hand side of (1) is modeled by assuming that if $\mathbf{s}$ is the speaker supervector for a randomly chosen speaker then

$$\mathbf{s} = \mathbf{m} + \mathbf{V}\mathbf{y} + \mathbf{D}\mathbf{z}, \tag{2}$$

where $\mathbf{m}$ is the speaker and channel independent supervector (UBM), $\mathbf{D}$ is a diagonal matrix, $\mathbf{V}$ is a rectangular matrix of low rank and $\mathbf{y}$ and $\mathbf{z}$ are independent random vectors having standard normal distributions. In other words, $\mathbf{s}$ is assumed to be normally distributed with mean $\mathbf{m}$ and covariance matrix $\mathbf{V}\mathbf{V}^* + \mathbf{D}\mathbf{D}^*$. The components of $\mathbf{y}$ and $\mathbf{z}$ are respectively the speaker and common *factors*.

The channel-dependent supervector $\mathbf{c}$, which represents the channel effect in an utterance, is assumed to be distributed according to

$$\mathbf{c} = \mathbf{U}\mathbf{x}, \tag{3}$$

where $\mathbf{U}$ is a rectangular matrix of low rank (known as eigenchannel matrix), $\mathbf{x}$ is a vector distributed with standard normal distribution. This is equivalent to saying that $\mathbf{c}$ is normally distributed with zero mean and covariance $\mathbf{U}\mathbf{U}^*$. The components of $\mathbf{x}$ are the channel factors in factor analysis modeling.

The underlying task in JFA is to train the hyperparameters $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{D}$ on a large training set. In the Bayesian framework, posterior distribution of the factors (knowing their priors) can be computed using the enrollment data. The likelihood of test utterance $\mathcal{X}$ is then computed by integrating over the posterior distribution of $\mathbf{y}$ and $\mathbf{z}$, and the prior distribution of $\mathbf{x}$ [6]. In [7], it was later shown, that using mere MAP point estimates of $\mathbf{y}$ and $\mathbf{z}$ is sufficient. Still, integration over the prior distribution of $\mathbf{x}$ was performed. We will further show, that using the MAP point estimate of $\mathbf{x}$ gives comparable results. Scoring is understood as computing the log-likelihood

[1] In the meaning of speaker verification

ratio (LLR) between the target speaker model $\mathbf{s}$ and the UBM, for the test utterance $\mathcal{X}$.

There are many ways in which JFA can be trained and which different sites have experimented with. Not only the training algorithms differ, but also the results were reported using different scoring strategies.

### 2.1. Frame by Frame

Frame-by-Frame is based on a full GMM log-likelihood evaluation. The log-likelihood of utterance $\mathcal{X}$ and model $\mathbf{s}$ is computed as an average frame log-likelihood [2]. It is practically infeasible to integrate out the channel, therefore MAP point estimate of $\mathbf{x}$ is used. The formula is as follows

$$\log P(\mathcal{X}|\mathbf{s}) = \sum_{t=1}^{T} \log \sum_{c=1}^{C} w_c \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c\right), \tag{4}$$

where $\mathbf{o}_t$ is the feature vector at frame $t$, $T$ is the length (in frames) for utterance $\mathcal{X}$, $C$ is number of Gaussians in the GMM, and $w_c$, $\boldsymbol{\Sigma}_c$, and $\boldsymbol{\mu}_c$ the $c$th Gaussian weight, mean, and covariance matrix, respectively.

### 2.2. Integrating over Channel Distribution

This approach is based on evaluating an objective function as given by Equation (13) in [2]:

$$P(\mathcal{X}|\mathbf{s}) \quad = \quad \int P(\mathcal{X}|\mathbf{s}, \mathbf{x}) \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{I}) \mathrm{d}\mathbf{x} \tag{5}$$

As was said in the previous paragraph, it would be difficult to evaluate this formula in the frame-by-frame strategy. However, (4) can be approximated by using fixed alignment of frames to Gaussians, i.e., assume that each frame is generated by a single (best scoring) Gaussian. In this case, the likelihood can be evaluated in terms of the sufficient statistics. If the statistics are collected in the Baum-Welch way, the approximation is equal to the GMM EM auxiliary function, which is a lower bound to (5). The closed form (logarithmic) solution is then given as:

$$\begin{aligned} \log \tilde{P}(\mathcal{X}|\mathbf{s}) \quad = \quad & \sum_{c=1}^{C} N_c \log \frac{1}{(2\pi)^{F/2} |\boldsymbol{\Sigma}_c|^{1/2}} \\ & - \frac{1}{2} \mathrm{tr}(\boldsymbol{\Sigma}^{-1} \mathbf{S}_\mathbf{s}) - \frac{1}{2} \log |\mathbf{L}| \\ & + \frac{1}{2} \| \mathbf{L}^{-1/2} \mathbf{U}^* \boldsymbol{\Sigma}^{-1} \mathbf{F}_\mathbf{s} \|^2 \end{aligned} \tag{6}$$

where for the first term, $C$ is the number of Gaussians, $N_c$ is the data count for Gaussian $c$, $F$ is the feature vector size, $\boldsymbol{\Sigma}_c$ is covariance matrix for Gaussian $c$. These numbers will be equal both for UBM and the target model, thus the whole term will cancel out in the computation of the log-likelihood ratio.

For the second term of (6), $\boldsymbol{\Sigma}$ is the block-diagonal matrix of separate covariance matrices for each Gaussian, $\mathbf{S}_\mathbf{s}$ is the second order moment of $\mathcal{X}$ around speaker $\mathbf{s}$ given as

$$\mathbf{S}_\mathbf{s} = \mathbf{S} - 2\mathrm{diag}(\mathbf{F}\mathbf{s}^*) + \mathrm{diag}(\mathbf{N}\mathbf{s}\mathbf{s}^*), \tag{7}$$

where $\mathbf{S}$ is the $CF \times CF$ block-diagonal matrix whose diagonal blocks are uncentered second order cumulants $\mathbf{S}_c$. This term is independent of speaker, thus will cancel out in the LLR computation (note that this was the only place where second order statistics appeared, therefore are not needed for scoring). $\mathbf{F}$ is a $CF \times 1$ vector, obtained by concatenating the first order statistics. $\mathbf{N}$ is a $CF \times CF$ diagonal matrix, whose diagonal blocks are $N_c \mathbf{I}_F$, i.e., the occupation counts for each Gaussian ($\mathbf{I}_F$ is $F \times F$ identity matrix).

The $\mathbf{L}$ in the third term of (6) is given as

$$\mathbf{L} = \mathbf{I} + \mathbf{U}^* \boldsymbol{\Sigma}^{-1} \mathbf{N} \mathbf{U}, \tag{8}$$

where $\mathbf{I}$ is a $CF \times CF$ identity matrix, $\mathbf{U}$ is the eigenchannel matrix, and the rest is as in the second term. The whole term, however, does not depend on speaker and will cancel out in the LLR computation.

In the fourth term of (6), let $\mathbf{L}^{1/2}$ be a lower triangular matrix, such that

$$\mathbf{L} = \mathbf{L}^{1/2} \mathbf{L}^{1/2*} \tag{9}$$

i.e., $\mathbf{L}^{-1/2}$ is the inverse of the Cholesky decomposition of $\mathbf{L}$.

As was said, terms one and three in (6), and second order statistics $\mathbf{S}$ in (7) will cancel out. Then the formula for the score is given as

$$\begin{aligned} Q_{\mathrm{int}}(\mathcal{X}|\mathbf{s}) \quad = \quad & \mathrm{tr}(\boldsymbol{\Sigma}^{-1} \mathrm{diag}(\mathbf{F}\mathbf{s}^*)) \\ & + \frac{1}{2} \mathrm{tr}(\boldsymbol{\Sigma}^{-1} \mathrm{diag}(\mathbf{N}\mathbf{s}\mathbf{s}^*)) \\ & + \frac{1}{2} \| \mathbf{L}^{-1/2} \mathbf{U}^* \boldsymbol{\Sigma}^{-1} \mathbf{F}_\mathbf{s} \|^2 \end{aligned} \tag{10}$$

### 2.3. Channel Point Estimate

This function is similar to the previous case, except for the fact, that the channel factor $\mathbf{x}$ is known. This way, there is no need for integrating over the whole distribution of $\mathbf{x}$, and only its point estimate is taken for LLR computation. The formula is directly adopted from [8] (Theorem 1),

$$\begin{aligned} \log \tilde{P}(\mathcal{X}|\mathbf{s}, \mathbf{x}) \quad = \quad & \sum_{c=1}^{C} N_c \log \frac{1}{(2\pi)^{F/2} |\boldsymbol{\Sigma}_c|^{1/2}} \\ & - \frac{1}{2} \mathrm{tr}(\boldsymbol{\Sigma}^{-1} \mathbf{S}) \\ & + \mathbf{M}^* \boldsymbol{\Sigma}^{-1} \mathbf{F} + \frac{1}{2} \mathbf{M}^* \mathbf{N} \boldsymbol{\Sigma}^{-1} \mathbf{M}, \end{aligned} \tag{11}$$

where $\mathbf{M}$ is given by (1). In this formula, the first and second terms cancel out in LLR computation, leading to scoring function

$$\begin{aligned} Q_{\mathrm{x}}(\mathcal{X}|\mathbf{s}, \mathbf{x}) \quad = \quad & \mathbf{M}^* \boldsymbol{\Sigma}^{-1} \mathbf{F} \\ & + \frac{1}{2} \mathbf{M}^* \mathbf{N} \boldsymbol{\Sigma}^{-1} \mathbf{M}, \end{aligned} \tag{12}$$

hence

$$\mathrm{LLR}_{\mathrm{x}}(\mathcal{X}|\mathbf{s}) = Q_{\mathrm{x}}(\mathcal{X}|\mathbf{s}, \mathbf{x}_\mathbf{s}) - Q_{\mathrm{x}}(\mathcal{X}|\mathrm{UBM}, \mathbf{x}_{\mathrm{UBM}}), \tag{13}$$

where $\mathbf{x}_{\mathrm{UBM}}$ is a channel factor estimated using UBM, and $\mathbf{x}_\mathbf{s}$ is a channel factor estimated using speaker $\mathbf{s}$.

### 2.4. UBM Channel Point Estimate

In [3], the authors assumed, that the shift of the model caused by the channel is identical both to the target model and the UBM[3]. Therefore, the $\mathbf{x}$ factor for utterance $\mathcal{X}$ is estimated using the UBM and then used for scoring. Formally written:

$$\begin{aligned} \mathrm{LLR}_{\mathrm{LPT}}(\mathcal{X}|\mathbf{s}) \quad = \quad & Q_{\mathrm{x}}(\mathcal{X}|\mathbf{s}, \mathbf{x}_{\mathrm{UBM}}) \\ & - Q_{\mathrm{x}}(\mathcal{X}|\mathrm{UBM}, \mathbf{x}_{\mathrm{UBM}}) \end{aligned} \tag{14}$$

[2] All scores are normalized by frame length of the tested utterance, therefore the log-likelihood is average.

[3] The authors identified themselves under abbreviation LPT, therefore we will refer to this approach as to LPT assumption

Note, that when computing the LLR, the $\mathbf{Ux}$ in the linear term of (11) will cancel out, leaving the compensation to the quadratic term of (11).

### 2.5. Linear Scoring

Let us keep the LPT assumption and let $\mathbf{m_c}$ be the channel compensated UBM:

$$\mathbf{m_c} \quad = \quad \mathbf{m} + \mathbf{c}. \tag{15}$$

Furthermore, let us assume, that we move the origin of supervector space to $\mathbf{m_c}$.

$$\bar{\mathbf{M}} \quad = \quad \mathbf{M} - \mathbf{m_c} \tag{16}$$

$$\bar{\mathbf{F}} \quad = \quad \mathbf{F} - \mathbf{N}\mathbf{m_c}. \tag{17}$$

Eq. (12) can now be rewritten to

$$\begin{aligned} Q_{\mathrm{xmod}}(\mathcal{X}|\bar{\mathbf{M}}, \mathbf{x}) \quad = \quad & \bar{\mathbf{M}}^*\mathbf{\Sigma}^{-1}\bar{\mathbf{F}} \\ & + \frac{1}{2}\bar{\mathbf{M}}^*\mathbf{N}\mathbf{\Sigma}^{-1}\bar{\mathbf{M}}. \end{aligned} \tag{18}$$

When approximating (18) by the first order Taylor series (as a function of $\bar{\mathbf{M}}$), only the linear term is kept, leading to

$$Q_{\mathrm{lin}}(\mathcal{X}|\bar{\mathbf{M}}, \mathbf{x}) \quad = \quad \bar{\mathbf{M}}^*\mathbf{\Sigma}^{-1}\bar{\mathbf{F}} \tag{19}$$

Realizing, that the channel compensated UBM is now a vector of zeros, and substituting (19) to (14), the formula for computing the LLR simplifies to

$$\mathrm{LLR}_{\mathrm{lin}}(\mathcal{X}|\mathbf{s}, \mathbf{x}) = (\mathbf{Vy} + \mathbf{Dz})^*\mathbf{\Sigma}^{-1}(\mathbf{F} - \mathbf{Nm} - \mathbf{Nc}). \tag{20}$$

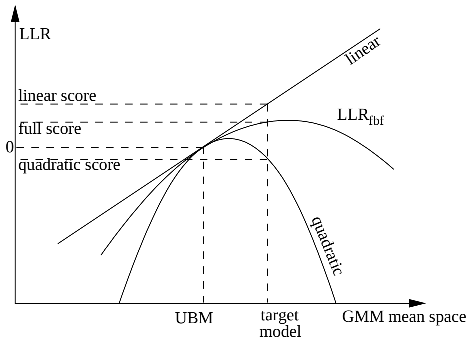


**Fig. 1**. An illustration of the scoring behavior for frame-by-frame, LPT, and linear scoring.

Given the fact, that the $\tilde{P}$-function is a lower bound approximation of the real frame-by-frame likelihood function, there are cases, when the LPT original function fails. Fig. 1 shows that the linear function can sometimes be a better approximation of the full LLR.

## 3. EXPERIMENTAL SETUP

### 3.1. Test Set

The results of our experiments are reported on the Det1 and Det3 conditions of the NIST 2006 speaker recognition evaluation (SRE) dataset [9].

The real-time factor was measured on a special test set, where 49 speakers were tested against 50 utterances. The speaker models were taken from the t-norm cohort, while the test utterances were chosen from the original z-norm cohort, each having approximately 4 minutes, totally giving 105 minutes.

### 3.2. Feature Extraction

In our experiments, we used cepstral features, extracted using a 25 ms Hamming window. 19 mel frequency cepstral coefficients together with log energy are calculated every 10 ms. This 20-dimensional feature vector was subjected to feature warping [10] using a 3 s sliding window. Delta and double delta coefficients were then calculated using a 5 frames window giving a 60-dimensional feature vectors. These feature vectors were modeled using GMM and factor analysis was used to treat the problem of speaker and session variability.

Segmentation was based on the BUT Hungarian phoneme recognizer [11] and relative average energy thresholding. Also short segments were pruned out, after which the speech segments were merged together.

### 3.3. JFA Training

We used gender independent Universal Background Models, which contain 2048 Gaussians. This UBM was trained using LDC releases of Switchboard II, Phases 2 and 3; switchboard Cellular, Parts 1 and 2 and NIST 2004-2005 SRE. The (gender independent) factor analysis models were trained on the same quantities of data as the UBM.

Our JFA is composed by 300 speaker factors, 100 channel factors, and diagonal matrix $\mathbf{D}$. While $\mathbf{U}$ was trained on the NIST data olny, $\mathbf{D}$ and $\mathbf{V}$ were trained on two disjoint sets comprising NIST and Switchboard data.

### 3.4. Normalization

All scores, as presented in the previous sections, were normalized by the number of frames in the test utterance. In case of normalizing the scores (zt-norm), we worked in the gender dependent fashion. We used 220 female, and 148 male speakers for t-norm, and 200 female, 159 male speakers for z-norm. These segments were a subset of the JFA training data set.

### 3.5. Hardware and Software

The frame-by-frame scoring was implemented in C++ code, which calls ATLAS functions for math operations. Matlab was used for the rest of the computations. Even though C++ produces more optimized code, the most CPU demanding computations are performed via the tuned math libraries that both Matlab and C++ use. This fact is important for measuring the real-time factor. The machine on which the real-time factor (RTF) was measured was a Dual-Core AMD Opteron 2220 with cache size 1024 KB. For the rest of the experiments, computing cluster was used.

## 4. RESULTS

Table 1 shows the results without any score normalization. The reason for the loss of performance in the case of LPT scoring could possibly be due to bad approximation of the likelihood function around UBM, ,i.e., the inability to adapt the model to the test utterance (in the $\mathbf{U}$ space only). Fig. 1 shows this case.

**Table 1**. *Comparison of different scoring techniques in terms of EER and DCF. No score normalization was performed here.*

| | Det1 | | Det3 | |
|---|---|---|---|---|
| | EER | DCF | EER | DCF |
| Frame-by-Frame | **4.70** | **2.24** | **3.62** | **1.76** |
| Integration | 5.36 | 2.46 | 4.17 | 1.95 |
| Point estimate | 5.25 | 2.46 | 4.17 | 1.96 |
| Point estimate LPT | 16.70 | 6.84 | 15.05 | 6.52 |
| Linear | 5.53 | 2.97 | 3.94 | 2.35 |

Table 2 shows the results after application of zt-norming. While the frame-by-frame scoring outperformed all the fast scorings in the un-normalized case, normalization is essential for the other methods.

**Table 2**. *Comparison of different scoring techniques in terms of EER and DCF. zt-norm was used as score normalization.*

| | Det1 | | Det3 | |
|---|---|---|---|---|
| | EER | DCF | EER | DCF |
| Frame-by-Frame | 2.96 | 1.50 | 1.80 | 0.91 |
| Integration | 2.90 | 1.48 | 1.78 | 0.91 |
| Point estimate | **2.90** | **1.47** | 1.83 | **0.89** |
| Point estimate LPT | 3.98 | 2.01 | 2.70 | 1.36 |
| Linear | 2.99 | 1.48 | **1.73** | 0.95 |

### 4.1. Speed

The aim of this experiment was to show the approximate real time factor of each of the systems. The time measured included reading necessary data connected with the test utterance (features, statistics), estimating the channel shifts, and computing the likelihood ratio. Any other time, such as reading of hyper-parameters, models, etc. was not comprised in the result. Each measuring was repeated 5 times and averaged. Table 3 shows the real time of each algorithm.

Surprisingly, the integration LLR is faster then the point estimate.

**Table 3**. *Real time factor for different systems*

| | Time [s] | RTF |
|---|---|---|
| Frame-by-Frame | 1010 | $1.60e^{-1}$ |
| Integration | 50 | $7.93e^{-3}$ |
| Point estimate | 160 | $2.54e^{-2}$ |
| Point estimate LPT | 36 | $5.71e^{-3}$ |
| Linear | **13** | $2.07e^{-3}$ |

This is due to implementation, where the channel compensation term in the integration formula is computed once per an utterance, while in the point estimate case, each model needs to be compensated for each trial utterance.

## 5. CONCLUSIONS

We have showed a comparison of different scoring techniques that different sites have recently used in their evaluations. While, in most cases, the performance does not change dramatically, the speed of evaluation is the major difference. The fastest scoring method is the Linear scoring. It can be implemented by a simple dot product, allowing for fast scoring of huge problems (e.g., z-, t- norming).

## 6. ACKNOWLEDGMENTS

This research was conducted under the auspices of the 2008 Johns Hopkins University Summer Workshop, and partially supported by NSF Grant No IIS-0705708 and by a gift from Google Inc. It was also partly supported by European projects AMIDA (FP6-033812) and MOBIO (FP7-214324), by Grant Agency of Czech Republic under project No. 102/08/0707, by Czech Ministry of Education under project No. MSM0021630528 and by Czech Ministry of Defense. The hardware used in this work was partially provided by CESNET under project No. 201/2006. Lukáš Burget was partly supported by Grant Agency of Czech Republic under project No. GP102/06/383.

## 7. REFERENCES

[1] Robert B. Dunn Douglas A. Reynolds, Thomas F. Quatieri, “Speaker verification using adapted gaussian mixture models,” *Digital Signal Processing*, pp. 19–41, January 2000.

[2] P. Kenny, G. Boulianne, P. Oullet, and P. Dumouchel, “Joint factor analysis versus eigenchannes in speaker recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

[3] C. Vair, D. Colibro, F. Castaldo, E. Dalmasso, and P. Laface, “Loquendo - politecnico di torino’s 2006 nist speaker recognition evaluation system,” in *Proceedings of Interspeech 2007*, 2007, pp. 1238–1241.

[4] Niko Brümmer, Lukáš Burget, Jan Černocký, Ondřej Glembek, František Grézl, Martin Karafiát, David Leeuwen van, Pavel Matějka, Petr Schwarz, and Albert Strasheim, “Fusion of heterogeneous speaker recognition systems in the stbu submission for the NIST speaker recognition evaluation 2006,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

[5] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, “A study of inter-speaker variability in speaker verification,” *IEEE Trans. Audio, Speech and Language Processing*, vol. 16, no. 5, pp. 980–988, July 2008.

[6] P. Kenny and P. Dumouchel, “Experiments in speaker verification using factor analysis likelihood ratios,” in *Proceedings of Odyssey 2004*, 2004.

[7] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, “Factor analysis simplified,” in *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Toulouse, France, March 2005, pp. 637– 640.

[8] P. Kenny, “Joint factor analysis of speaker and session variability : Theory and algorithms - technical report CRIM-06/08-13. Montreal, CRIM, 2005,” 2005.

[9] “National institute of standard and technology,” http://www.nist.gov/speech/tests/spk/index.htm.

[10] S. Sridharan J. Pelecanos, “Feature warping for robust speaker verification,” in *Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop*, 2006, pp. 213–218.

[11] P. Schwarz, P. Matějka, and J. Černocký, “Hierarchical structures of neural networks for phoneme recognition,” in *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Toulouse, France, May 2006, pp. 325–328.

# Discriminative Training and Channel Compensation for Acoustic Language Recognition

*Valiantsina Hubeika, Lukáš Burget, Pavel Matějka, Petr Schwarz*

Speech@FIT, Brno University of Technology, Czech Republic
xhubei00@stud.fit.vutbr.cz, {burget|matejkap|schwarzp}@fit.vutbr.cz

## Abstract

This paper describes the acoustic language recognition subsystems of Brno University of Technology (BUT) which contributed to the BUT main submission to the NIST LRE 2007. Two main techniques are employed in the subsystems discriminative training in terms of Maximum Mutual Information, and channel compensation in terms of eigenchannel adaptation in both, model and feature domain. The complementarity of the approaches is analyzed.

**Index Terms**: Language detection, NIST LRE 2007 evaluation, discriminative training, eigenchannel adaptation in model domain, eigenchannel adaptation in feature domain

## 1. Introduction

To date, there is a fair number of methods developed to improve performance of the state-of-the-art acoustic language recognition systems. Still, two issues are main challenges in the task, inter-session channel variability compensation as recordings belonging to the same language may be obtained through different channels, and language discrimination as some languages may have common features. This paper addressed both these problems within the UBM-GMM framework [12]. Here, to compensate on the channel, eigenchannel adaptation technique is applied; to train the models descriptively, Maximum Mutual Information (MMI) is used.

Formerly, a channel compensation method was proposed task by Kenny [22] in terms of factor analysis (FA). Brümmer [13] has developed a simplified version of FA, eigenchannel adaptation. These methods were developed within GMM framework and are implemented in model domain. Later, Castaldo in [7] has introduced an approximation of eigenchannel adaptation, eigenchannel adaptation in feature domain. With channel compensation performed in feature domain, different approaches can be used for the feature distribution modeling. Both compensating techniques, eigenchannel adaptation in model and feature domain, were involved in our systems.

As was proven during LRE 2005 in [2], discriminative training, by means of MMI, in language recognition task is highly beneficial and brought a great decrease in EER.

We investigate improvements given by both approaches and their combination. Further, we examine complementarity of the both methods and systems based on approaches of different nature, such as phonotactic systems.

## 2. Theoretical Background

This section gives a brief information on the objectives of eigenchannel adaptation and discriminative training.

### 2.1. Eigenchannel Adaptation in Model Domain

Let supervector be a $MD$ dimensional vector constructed by concatenating all GMM mean vectors and normalized by corresponding standard deviations. $M$ is the number of Gaussian mixture components in GMM and $D$ is dimensionality of features. Before eigenchannel adaptation can be applied, we must identify directions in which supervector is mostly affected by changing channel. These directions (eigenchannels) are defined by columns of $MD \times R$ matrix $\mathbf{V}$, where $R$ is the chosen number of eigenchannels ($R = 50$ in our system). The matrix $\mathbf{V}$ is given then by $R$ eigenvectors of average within-class covariance matrix, where each class is represented by supervectors estimated on different segments of the same language.

Once the eigenchannels are identified, language-dependent model (or language-independent UBM) can be adapted to a test conversation by shifting its supervector in the directions given by eigenchannels to better fit the test conversation data. Mathematically, this can be expressed as finding the channel factors, $x$, that maximize the following MAP criterion:

$$p(\mathbf{O}|\mathbf{s} + \mathbf{V}\mathbf{x})N(\mathbf{x}; \mathbf{0}, \mathbf{I}) \tag{1}$$

where $\mathbf{s}$ is supervector representing the model to be adapted, $p(\mathbf{O}|\mathbf{s} + \mathbf{V}\mathbf{x})$ is likelihood of the test conversation given the adapted supervector (model) and $N(\mathbf{x}; \mathbf{0}, \mathbf{I})$ denotes normally distributed vector. Assuming fixed occupation of Gaussian mixture components by test conversation frames, $\mathbf{o}_t, t = 1, \ldots, T$, it can be shown [13] that $\mathbf{x}$ maximizing criterion (1) is given by:

$$\mathbf{x} = \mathbf{A}^{-1} \sum_{m=1}^{M} \mathbf{V}_m^T \sum_{t=1}^{T} \gamma_m(t) \frac{\mathbf{o}_t - \mu_m}{\sigma_m} \tag{2}$$

where $\mathbf{V}_m$ is $D \times R$ part of matrix $\mathbf{V}$ corresponding to $m^{th}$ mixture component, $\gamma_m(t)$ is the probability of occupation mixture component $m$ at time $t$, $\mu_m$ and $\sigma_m$ are the mixture component's mean and standard deviation vectors of the model to be adapted and

$$\mathbf{A} = \mathbf{I} + \sum_{m=1}^{M} \mathbf{V}_m^T \mathbf{V}_m \sum_{t=1}^{T} \gamma_m(t). \tag{3}$$

In our implementation, occupation probabilities, $\gamma_m(t)$, are computed using UBM and assumed to be fixed for given test conversation.

### 2.2. Eigenchannel Adaptation in Feature Domain

Adaptation in feature domain aims at projecting every observation feature $\mathbf{o}(t)$ to the session-independent space. Channel factors, $\mathbf{x}$, are estimated using UBM (and not speaker-dependent models). The adapted feature vector is then obtained using 1-best Gaussian in the following way:

$$\mathbf{o}_t^{'} = \mathbf{o}_t + \mathbf{V}_m \mathbf{x} \tag{4}$$

where $m$ is the index of the best scored Gaussian and $\mathbf{V}_m$ is the part of $\mathbf{V}$ corresponding to the $m$-th Gaussian.

### 2.3. Maximum Mutual Training

Unlike in the case of ML training which aims to maximize the overall likelihood of training data given the transcriptions, the MMI objective function to maximize is the posterior probability of correctly recognizing all training segments:

$$F_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{p_\lambda(O_r|s_r)^{K_r} P(S_r)}{\sum_{\forall s} p_\lambda(O_r|s)^{K_r} P(s)} \tag{5}$$

where $p_\lambda(O_r|s_r)$ is likelihood of $r$-th training segment, $O_r$, given the correct transcription of the segment, $s_r$, and model parameters, $\lambda$. $R$ is the number of training segments and the denominator represents the overall probability density, $p_\lambda(O_r)$. Definition of the re-estimation formula is to be found in [2].

## 3. Experimental Setup

The results are presented in terms of the $100 \times C_{avg}$ (the formulas are to be found in [17]).

### 3.1. Data

*3.1.1. Training Data*

To compile the training data set, different sources were used (NIST1996, NIST2003, NIST2005, CallHome, CallFriend, Fisher, Mixer, OGI-multilingual, OGI 22 languages, Foreigen Accented Englis, SpeechDat-East) [20]. The amount of training data for different languages greatly varied, from 1.5h for Thai language to 228h for English.

The training data was divided onto two subsets: the first subset was used for training the models of languages and the second was used for training of the back-end parameters.

*3.1.2. Evaluation data*

NIST LRE2007 data was used as the evaluation data. There are 14 languages defined as detection targets with more than 7500 segments to identify. The evaluation set contains test segments with three nominal durations of speech: 3, 10 and 30 seconds. Detailed information can be found in the NIST LRE 2007 evaluation plan [17].

### 3.2. Systems

*3.2.1. Pre-processing*

The voice activity detection (VAD) is performed by our Hungarian phoneme recognizer [15], with all the phoneme classes linked to 'speech' class. The frames containing silence are excluded from the further processing.

*3.2.2. Features*

All systems use the shifted-delta-cepstra (SDC) [1] together with direct MFCC. The feature extraction was the same as in our LRE 2005 system [2]: 7 MFCC coefficients (including coefficient C0) concatenated with SDC 7-1-3-7, which totals in 56 coefficients per frame.

The features were transformed using vocal-tract length normalization (VTLN) [5]. The warping factors are estimated using single GMM (512 Gaussians), ML-trained on the whole CallFriend database (using all the languages). The model was trained in standard speaker adaptive training (SAT) fashion in four iterations of alternately re-estimating the model parameters and the warping factors for the training data.

*3.2.3. GMM system with 2048 Gaussians per language with eigenchannel adaptation in model domain:* `GMM2048-eigchan`

The inspiration comes from our GMM system for speaker recognition [14] which follow conventional Universal Background Model-Gaussian Mixture Modeling (UBM-GMM) paradigm [12].

Each language-dependent model is obtained by traditional *relevance MAP* adaptation [4] of UBM using enrollment conversation. Only the means are adapted with the relevance factor $\tau = 19$.

In the verification phase, standard Top-N Expected Log Likelihood Ratio (ELLR) scoring [4] is used to obtain verification score, where $N = 10$ in our system. However, for each trial, both the language-dependent model and the UBM are adapted to the channel of the test conversation using eigenchannel adaptation in model domain prior to computing the log likelihood ratio score.

The eigenchannel matrix was composed of eigenchannels derived in the following way:

1. UBM is trained using the original features.
2. For each utterance, a new GMM is obtained by MAP adaptation.
3. A supervector of means normalized by corresponding standard deviations is obtained from each GMM.
4. A maximum of 100 supervectors per database and language were selected.
5. The mean is subtracted from supervectors over each language of a database (not over language as one would expect)
6. Eigenchannels (i.e. directions in which language-dependent models are adapted for each test utterance) are given by eigen vectors of the covariance matrix estimated from the supervectors (see [3] for details).

*3.2.4. GMM system with 2048 Gaussians per language with eigenchannel adaptation in feature domain* `GMM2048-chcf`

A similar set of GMM models with 2048 Gaussians per language was trained in UBM-GMM fashion. However, the features (both, the training and test set) were first compensated using eigenchannel adaptation in feature domain [10, 11] (where eigenchannel matrix was the same as in the standard approach, see 3.2.3). In the case of the training data, the channel factors (see equation 1) were estimated using the UBM with 2048 Gaussians. The test data was channel compensated in the same manner as the training data. However, due to the short duration of the segments, to achieve better generalization (as eigenchannels can be estimated more robustly from the covariance matrix), the UBM with 256 Gaussians was used for channel factor estimation.

Table 1: Performance of our acoustic systems on LRE 2007 data

| | 30 sec | 10sec | 3sec |
|---|---|---|---|
| GMM2048, baseline | 8.03 | 12.89 | 21.77 |
| GMM2048-eigchan | 2.76 | 7.38 | 17.14 |
| GMM2048-chcf | 2.94 | 7.40 | 17.93 |
| GMM256-MMI ( 15 MMI it) | 4.15 | 8.61 | 18.43 |
| GMM256-MMI-chcf ( 3 MMI it) | 3.73 | 9.81 | 20.98 |
| GMM2048-MMI-chcf ( 3 MMI it) | 2.41 | 7.02 | 16.90 |

Table 2: Performance of our best-performing acoustic and phonotactic system, and their fusion

| | 30 sec | 10sec | 3sec |
|---|---|---|---|
| (1) GMM2048-MMI-chcf | 2.41 | 7.02 | 16.90 |
| (2) EN_Tree | 3.54 | 10.69 | 22.66 |
| (1) + (2) (LDA fusion) | 1.50 | 5.27 | 14.55 |

#### 3.2.5. *GMM-MMI:* `GMM256-MMI`

This system uses GMM models with 256 Gaussians per language as the base models, where mean and variance parameters were iteratively re-estimated using Maximum Mutual Information criterion - the same as for LRE2005 [2]. A relatively small number of Gaussians was chosen for high resource consumption during MMI training. The models' parameters were re-estimated in 15 iterations.

#### 3.2.6. *GMM-MMI with channel compensated features:* `GMM256-MMI-chcf`, `GMM2048-MMI-chcf`

The GMM256-MMI-chcf system was trained in an identical manner as the GMM256-MMI system, however the features were preliminary compensated by means of eigenchannel adaptation in feature domain.

In the GMM2048-MMI-chcf system the number of Gaussians per language was increased to 2048.

### 3.3. Normalization and Calibration

In this work, all results are presented for the systems calibrated using linear Gaussian back-end (LDA) and linear logistic regression back-end (LLR) [8] used in cascade. During LDA, for each class, a single full-covariance Gaussian (the covariance matrix is shared among all classes) is trained on the vector of scores generated from all models. LLR is trained in a discriminative fashion. The FoCal Multi-class toolkit by Niko Brummer[1] was used for this purpose.

## 4. Results

We used a UBM-GMM system with 2048 Gaussians per language as the baseline system, where no eigenchannel adaptation was employed (GMM2048). Results of the individual systems described above and the baseline are listed in Table 1.

When eigenchannel adaptation in model domain was applied, GMM2048-eigchan, the error decreased almost to one third of the baseline. When eigenchannel adaptation was done in feature domain, GMM2048-chcf, the error was slightly higher than for GMM2048-eigenchan but the approach enables simple application of additional MMI parameter re-training to improve the performance.

Table 3: Effect of calibration for the GMM2048-MMI-chcf on LRE 2007 data

| | 30 sec | 10 sec | 3 sec |
|---|---|---|---|
| No back-end | 5.75 | 9.45 | 18.44 |
| LDA+LLR | 2.41 | 7.02 | 16.90 |

Then several experiments were run by applying MMI training in order to select the best performing configuration. Inspired by our 2005 LID system, GMM-MMI system was first trained with 256 Gaussians. In this case, 15 iterations of the parameter re-estimations were required to converge. the error of this system was significantly lower than the error of the baseline, however the system did not reach the performance of the GMM2048-eigchan system.

Observing the good performance of the systems employing eigenchannel adaptation and MMI training, respectively, and assuming complementarity of the techniques, our intention was to combine both techniques in order to achieve further improvement of the result. When the models with 256 Gaussians were trained on the compensated features and the parameters of the models were re-estimated by means of MMI, where already 3 iterations were sufficient, we observed relative improvements of 22 % to the accuracy of the GMM256-MMI system on 30 sec condition.

Still, we supposed there was room for further improving of the recognition by increasing the number of Gaussians. When the models were trained in the same manner as GMM256-MMI-chcf only with the number of Gaussians increased to 2048 (again, only 3 iterations were run), the system out-performed the 2048GMM-eigchan system by 35 % relative in 30 sec condition.

### 4.1. Calibration

The calibration of the obtained scores was an important part in building our systems. To outline the effect of the calibration, the results of the uncalibrated GMM2048-MMI-chcf system are present as well as of the calibrated system (see Tab 3). However, in case of 3 sec condition, the decrease of the error is only about 8 % relative, in case of 30 sec condition, we could observe more than 50 % of relative reduction of the error.

### 4.2. Complementarity with the Other System

In order to draw an overview of the performance of our acoustic systems, we present (for sake of comparison) results achieved with our best phonotactic system, EN_Tree (see Tab 2) [21]. The approach is based on recognizing of the phonemes using English phoneme recognizer and following language modeling (PRLM). The EN_Tree system employs binary decision tree language modeling based on creating a single language independent tree (UBM) and adapting its distributions to individual language training data, as described in Navratil's work [18, 19]. Binary decision tree is trained on posterior weighted counts from phoneme lattices [2]. When both, our best-performing acoustic system GMM2048-chcf and EN_Tree, were fused, we observed a great reduction in ERR which indicates high com-

[1] http://niko.brummer.googlepages.com/focalmulticlass

plementarity of the systems. Complementarity of our other systems was further examined, for a detailed description see [20].

## 5. Conclusion

We showed that both eigenchannel adaptation and MMI training are greatly beneficial in the language recognition task. It was shown that, the approximation of the standard eigenchannel adaptation, eigenchannel adaptation in feature domain is almost as accurate as the standard approach. Moreover, it has a great advantage, that it allows to apply MMI parameter re-estimation without modifying the MMI training algorithm. We showed that when eigenchannel adaptation is applied in feature domain, further improvement of the result can be achieved by subsequent re-estimating of the parameter of GMM by using MMI training. We showed that our best acoustic system is complementary and well fused with our other systems. We have also shown, the calibration of the obtained scores is an important part of building an accurate recognition system.

## 6. Acknowledgment

This work was partly supported by European projects AMIDA (FP6-033812), Caretaker (FP6-027231) and MOBIO (FP7-214324), by Grant Agency of Czech Republic under project No. 102/08/0707 and by Czech Ministry of Education under project No. MSM0021630528. The hardware used in this work was partially provided by CESNET under project No. 201/2006. Lukáš Burget was supported by Grant Agency of Czech Republic under project No. GP102/06/383.

## 7. References

[1] P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, and J.R. Deller Jr., “Approaches to language identification using Gaussian mixture models and shifted delta cepstral features,” in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sept. 2002, pp. 89–92.

[2] P. Matějka, L. Burget, P. Schwarz, and J. Černocký, “Brno University of Technology system for NIST 2005 Language recognition evaluation,” in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006, pp. 57–64.

[3] L. Burget, P. Matejka, P. Schwarz, O. Glembek, and J. Cernocky: Analysis of feature extraction and channel compensation in GMM speaker recognition system, In: IEEE Transactions on Audio, Speech, and Language Processing, Vol. 15, No. 7, 2007, pp. 1979-1986, ISSN 1558-7916.

[4] D. A. Reynolds, “Comparison of background normalization methods for text-independent speaker verification,” in *Proc. Eurospeech*, Rhodes, Greece, Sept. 1997, pp. 963–966.

[5] J. Cohen, T. Kamm, and A.G. Andreou, “Vocal tract normalization in speech recognition: Compensating for systematic speaker variability,” *J. Acoust. Soc. Am.*, , no. 97, pp. 2346, 1995.

[6] E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, and D.A. Reynolds, “Acoustic,phonetic,and discriminative approaches to automatic language identification,” in *Proc. Eurospeech*, Sept. 2003, pp. 1345–1348.

[7] Castaldo, F.. Colibro, D.. Dalmasso, E.. Laface, P.. Vair, C.. “Compensation of nuisance factors for speaker and language recognition”, In: IEEE Transactions on Audio, Speech, and Language Processing, Vol. 15, No. 7, 2007, pp 1969–1978, ISSN 1558-7916.

[8] N. Brmmer, L. Burget, J. Cernocky, O. Glembek, F. Grezl, M. Karafiát, D. van Leeuwen, P. Matejka, P. Schwarz, and A. Strasheim, “Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 2072–2084, Sept. 2007.

[9] W. Campbell, D.E. Sturim, D.A. Reynolds, and A. Solomonoff, “Svm based speaker verification using a GMM supervector kernel and nap variability compensation,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, vol. I, pp. 97–100.

[10] V. Hubeika, L. Burget, P. Matejka, and J. Cernocky, “Channel compensation for speaker recognition,” in *Proc. Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, June 2007.

[11] F. Castaldo, E. Dalmasso, P. Laface, D. Colibro, and C. Vair, “Language identification using acoustic models and speaker compensated cepstral-time matrices,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, Oct. 2007, vol. 4, pp. 1013–1016.

[12] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted Gaussian mixture models,” *Digital Signal Processing*, vol. 10, no. 1–3, pp. 19–41, 2000.

[13] Niko Brummer, “Spescom DataVoice NIST 2004 system description,” in *Proc. NIST Speaker Recognition Evaluation 2004*, Toledo, Spain, June 2004.

[14] L. Burget, P. Matejka, P. Schwarz, O. Glembek, and J. Cernocky, “Analysis of feature extraction and channel compensation in GMM speaker recognition system,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 1979–1986, Sept. 2007.

[15] P. Schwarz, P. Matějka, and J. Černocký, “Towards lower error rates in phoneme recognition,” in *Proc. International Conference on Text, Speech and Dialogue*, Brno, Czech Republic, Sept. 2004, pp. 465–472.

[16] W. Campbell, D. Sturim, and D. Reynolds, “Support vector machines using GMM supervectors for speaker verification,” *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[17] NIST 2007 language recognition evaluation plan (lre07) www.nist.gov/speech/tests/lang/2007/lre07evalplanv8b.pdf

[18] J. Navratil: Spoken language recognition-a step toward multilinguality in speech processing, in IEEE Trans. on Speech and Audio Processing, Vol. 9, No. 6, pp. 678-685 ISSN: 1063-6676, September 2001.

[19] J. Navratil: “Recent advances in phonotactic language recognition using binary-decision trees,” in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Pittsburgh, PA, October 2006

[20] P. Matějka at al., “Brno university of technology system for nist 2007 language recognition evaluation,” in submited to: *Proc. International Conferences on Spoken Language Processing (ICSLP), Brisbane, Australia*

[21] O. Glembek et al.: Advances in phonotactic language recognition, submitted to *Proc. International Conferences on Spoken Language Processing (ICSLP), Brisbane, Australia.*

[22] P. Kenny, P. Dumouchel (2004): “Experiments in speaker verification using factor analysis likelihood ratios”, in *Odyssey: The Speaker and Language Recognition Workshop* , 2004 , pp. 219–226.

Available online at www.sciencedirect.com

ScienceDirect

Speech Communication 53 (2011) 1172–1185

SPEECH COMMUNICATION

www.elsevier.com/locate/specom

# Application of speaker- and language identification state-of-the-art techniques for emotion recognition ☆

Marcel Kockmann *, Lukáš Burget, Jan "Honza" Černocký

*Brno University of Technology, Speech@FIT, Czech Republic*

Available online 1 February 2011

**Abstract**

This paper describes our efforts of transferring feature extraction and statistical modeling techniques from the fields of speaker and language identification to the related field of emotion recognition. We give detailed insight to our acoustic and prosodic feature extraction and show how to apply Gaussian Mixture Modeling techniques on top of it. We focus on different flavors of Gaussian Mixture Models (GMMs), including more sophisticated approaches like discriminative training using Maximum-Mutual-Information (MMI) criterion and InterSession Variability (ISV) compensation. Both techniques show superior performance in language and speaker identification. Furthermore, we combine multiple system outputs by score-level fusion to exploit the complementary information in diverse systems. Our proposal is evaluated with several experiments on the FAU Aibo Emotion Corpus containing non-acted spontaneous emotional speech. Within the Interspeech 2009 Emotion Challenge we could achieve the best results for the 5-class task of the Open Performance Sub-Challenge with an unweighted average recall of 41.7%. Further additional experiments on the acted Berlin Database of Emotional Speech show the capability of intersession variability compensation for emotion recognition.

*Keywords:* Emotion recognition; Gaussian mixture models; Maximum-mutual-information; Intersession variability compensation; Score-level fusion

## 1. Introduction

Spoken emotion recognition is the problem of automatically recognizing the emotional state of a person from their speech. Different moods may change the attributes of the human voice, such as pitch, speaking-rate, and intonation.

In automatic speech processing these properties are usually represented using the appropriate parametrization of speech, so called features. Pattern recognition and machine learning algorithms can then be used to model certain characteristics of emotionally colored speech and recognize emotions in speech utterances. Typically, classifiers like Hidden-Markov-Models (HMMs), Gaussian Mixture Models (GMMs), Support Vector Machines (SVMs) or Neural Networks (NNs) (Bishop, 2006) are used.

While sensing the emotions of an individual from their speech is a relatively new research field in speech processing, a research community has formed in recent years and several methods have been applied successfully (Steidl, 2009; Vlasenko et al., 2007; Seppi et al., 2008; Batliner et al., 2006) and evaluated on special databases containing emotional speech (Ververidis and Kotropoulos, 2003).

Recently the usage of SVMs to directly model large-scale feature vectors has become the standard for emotion recognition (Schuller et al., 2007, 2009). These feature vectors contain diverse kinds of speech parametrization extracted on a per-utterance basis including acoustic, prosodic and voice quality features. Frame based features are usually modeled by HMMs to capture the temporal dynamics of the signal (Schuller et al., 2009).

Using these state-of-the-art techniques, accuracies of over 80% have been reported for emotion classification

☆ This work was partly supported by European project MOBIO (FP7-214324), by Grant Agency of Czech Republic project No. 102/08/0707, and by the Czech Ministry of Education project No. MSM0021630528. Marcel Kockmann is supported by SVOX Deutschland GmbH, Munich, Germany.

* Corresponding author.
*E-mail address:* kockmann@fit.vutbr.cz (M. Kockmann).

0167-6393/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.specom.2011.01.007

tasks on acted non-spontaneous data (Schuller et al., 2006). However, on real life non-acted spontaneous emotionally colored data these accuracies drop drastically (below 40%) (Schuller et al., 2009).

Besides emotion recognition there are many diverse research fields with the goal of extracting certain attributes from speech. These include:

- What is spoken: Automatic Speech Recognition (ASR).
- Who is speaking: Speaker Identification (SID).
- Which language is used: Language Identification (LID).
- Which gender is the speaker: Gender identification (GID).
- What is the age of the speaker: Age Identification (AID).

In many of these fields (like SID, LID and GID) the use of Gaussian Mixture Models has established itself as the standard (Reynolds et al., 2000). HMMs, as used in ASR, are usually outperformed by GMMs (which are actually a HMM containing a single state) on text-independent tasks. Also, best results in all these fields are often obtained using more or less standard acoustic features extracted on a frame-based level, as used in ASR. This is somewhat illogical as features for ASR are optimized to blind out properties like speaker characteristics. Still, these tools seem to provide a good framework for diverse kinds of speech characterization.

As mentioned above, the state-of-the-art for emotion recognition has moved in a different direction. Gaussian mixture modeling of short-time acoustic features has been mostly replaced by Support Vector Machine classification. A similar trend was observed in the field of Speaker Verification as well. However, recent advances in Gaussian Mixture Modeling, like discriminative training or intersession variability compensation, has significantly raised the performance of GMM based systems and currently defines the state-of-the-art (Kinnunen and Li, 2010). This is the main motivation for our work. Our aim is to take basic and newly evolved features and modeling techniques, as used in current LID and SID systems and to apply them to the task of emotion recognition. By doing so we want to provide another view to the problem of emotion recognition. Further enhancement can then be expected by combining both approaches.

Through this paper, we will investigate standard spectral features based on Mel-Frequency-Cepstral-Coefficients (MFCC) (Davis and Mermelstein, 1980) as they are usually used in ASR. There have been many modifications of standard MFCC features to better fit the needs of SID and LID, like longer temporal context and speaker normalization. We will evaluate below some of these techniques for emotion recognition.

Furthermore, prosodic features (incorporating duration, pitch and energy) are often used to enhance the performance of MFCC based systems. Different from spectral features, prosodic features are usually extracted over a longer time span, like on a syllable basis. We examined a prosodic feature extraction method successfully used for GMM based speaker recognition (Kockmann and Burget, 2008).

All these features will be modeled using different flavors of Gaussian Mixture Models. It should be noted, that in all cases we model frame or syllable based features using models without any temporal dependencies. This statistical method of creating a “footprint” has been very successful. We will investigate in detail basic GMM approaches used in speaker and language identification. Furthermore, more sophisticated techniques evolved in the last few years are examined for their applicability in emotion recognition. These include discriminative training of GMMs and intersession variability compensation. Intersession variability for emotion recognition may refer to different acoustic conditions, different speakers or simply the spoken content of the utterance. All these attributes are a nuisance for the task of emotion recognition and we want to “ignore” them during modeling.

To evaluate the performance of the proposed techniques we provide experiments on two independent emotional databases, one containing non-acted spontaneous speech and the other acted non-spontaneous speech. Results on the first database include our submission to the Interspeech 2009 Emotion Challenge (Kockmann et al., 2009) where we could achieve very good results using the techniques described above.

The paper is organized as follows: Section 2 describes the acoustic features we used in our experiments while Section 3 explains the prosodic features used. Section 4 gives detailed information on the Gaussian Mixture Models we used and their training and evaluation procedures. In Sections 5, 6 we present results to evaluate the proposed approaches for emotion recognition. In Section 7 we draw conclusions to our approaches and consider future research.

## 2. Spectral features

This section will introduce the used MFCC features and the additional techniques applied to make them more suitable for the given task.

### 2.1. Basic acoustic features

The most widely used features in speech processing are MFCCs (Davis and Mermelstein, 1980). They have been applied successfully for speech recognition as well as for speaker recognition and language identification. We will use them as our basic features for the emotion recognition task. MFCC vectors are generated every 10 ms on a 20 ms frame of speech weighted by a Hamming window. Fast-Fourier-Transform (FFT) output of each speech window is processed by a Mel filter bank with 25 bands. The output is transformed by Discrete Cosine Transform (DCT) and

13 cepstral coefficients including C0 are generated. C0 represents an energy measure of the speech window.

### 2.2. Channel normalization

The temporal trajectories of individual cepstral coefficients are filtered using a standard RelAtive SpecTrAl (RASTA) filter (Hermansky and Morgan, 1994) to remove slow and very fast spectral changes which do not appear to be characteristic for natural speech. We use the standard IIR filter:

$$H(z) = 0.1\frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{z^{-4}(1 - 0.982z^{-1})}. \tag{1}$$

Furthermore, cepstral mean subtraction (CMS) is applied on each coefficient per utterance for simple channel normalization.

### 2.3. Speaker normalization

We do not want to model the characteristics of the individual speaker by the position of the formants based on the length of the vocal tract. We use Vocal Tract Length Normalization (VTLN) (Cohen et al., 1995) for simple speaker normalization. The spectrum (during FFT) is either compressed (usually for females) or expanded (for male speakers) based on a warping factor estimate for each utterance.

Warping factors for training and test data are estimated using a rather small GMM trained on all unnormalized training data to represent average characteristics of the target population. Warped MFCCs are then created for all files with warping factors in a range from 0.88–1.12 with a step-size of 0.02. This results in 13 feature sets: 6 compressed, 1 neutral and 6 expanded. The optimal warping factor per utterance is obtained by evaluating the likelihood of all warped instances against the unnormalized GMM and selecting the maximum. This way we select the factor that best fits the average speaker. The warped utterances are then used for standard model training. Refer to Section 4.1 for implementation details for GMM training and likelihood scoring. For spectrum manipulation we use a linear piecewise warping function with a warping cutoff of $0.875 \times N_f$, where $N_f$ is the Nyquist frequency.

### 2.4. Temporal context

Simple MFCCs do not model any temporal characteristics which are most likely informative for emotion recognition. As our classifier also does not model feature sequences, we generate delta, double and triple delta regression coefficients of the static features to model coarticulations in speech. We use a standard formula (Young et al., 2006):

$$d_t = \frac{\sum_{\forall\Theta}\Theta(c_{t+\Theta} - c_{t-\Theta})}{2\sum_{\forall\Theta}\Theta^2} \tag{2}$$

with $d_t$ being the regression coefficient of static coefficient $c_t$ and the shift vector $\Theta = [2]$ for delta, $\Theta = [2,4]$ for double delta and $\Theta = [2,4,6]$ for triple deltas. This results in 26, 39 and 52 dimensional feature vectors containing information spanning a context of 5, 9 and 13 frames, respectively.

### 2.5. Shifted delta cepstra

The importance of an even broader temporal information has been shown for LID (Torres-Carrasquillo et al., 2002). The so-called Shifted Delta Cepstra (SDC) is created by stacking delta coefficients computed across multiple speech frames, as depicted in Fig. 1. Multiple delta coefficients with a shift of ±1 are computed for a context of ±10 frames, without overlap and concatenated in one feature vector.

For static features $c_t$ shifted deltas are defined:

$$\Delta c_t = c_{(t+iP+d)} - c_{(t+iP-d)} \tag{3}$$

for $i = [-3\ldots0\ldots3]$ with shift $P = 3$ and the window shift $d = 1$ over which deltas are computed.

The basic features in our system are 7 static MFCC coefficients (including coefficient C0) concatenated with delta cepstra which totals 56 SDC coefficients per frame, spanning a context of 21 frames. This configuration has been successfully used in our language identification systems (Matejka et al., 2008, 2006).

### 2.6. Post processing: voice activity detection

For all our frame based spectral features, non-speech frames are discarded and only speech frames are considered in the following stages of training models and verification. Speech/non-speech segmentation is performed by our Hungarian phone recognizer (Schwarz et al., 2006). This step is performed based on the final feature vectors ensuring that RASTA and regression coefficients are correctly estimated.

## 3. Prosodic features

Prosodic information based on the lexical context might be useful for this task and is complementary to the acoustic short time features. For this purpose, we use our detector of syllable-based feature contours as presented in (Kockmann and Burget, 2008). It processes classical prosodic features like duration, pitch and energy in a syllable-like temporal context. The trajectories of each feature are continuously modeled over the time span of a syllable and are represented by discrete cosine transformation (DCT) coefficients, as depicted in Fig. 2. The pseudo-syllable segmentation is based on a phone recognizer where vowels are considered as nuclei for the syllables. The segments are non-overlapping and undefined frames are discarded prior to DCT approximation. Additionally, we also capture the temporal contours of MFCCs and form a single feature vector out of duration, pitch, energy and

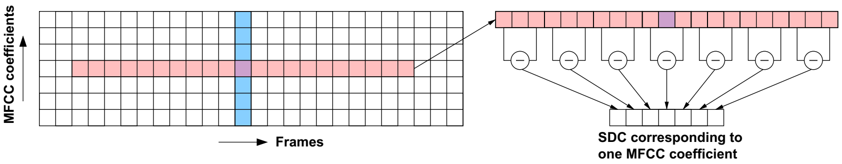


Fig. 1. Computation of SDC features for a single static feature stream, incorporating 21 consecutive static MFCCs, results in 7-dimensional SDC vector for each frame.

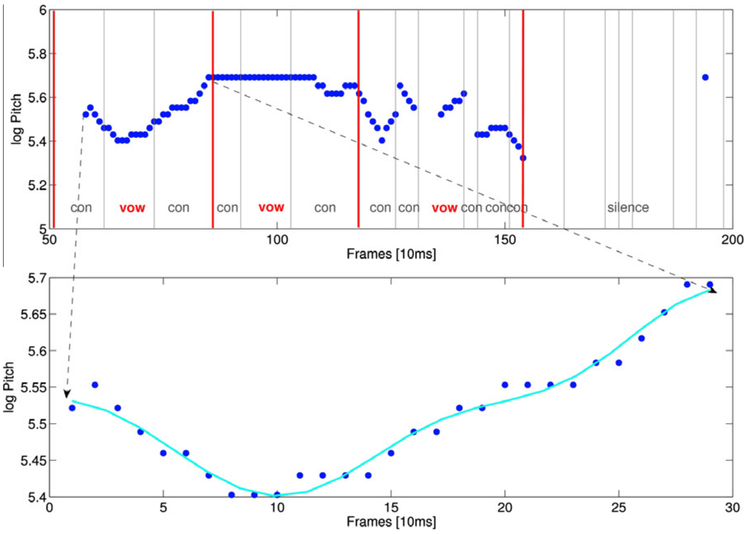


Fig. 2. Example of a pitch contour over a syllable consisting of three phones. Top: Original pitch values with phone and pseudo-syllable boundaries (horizontal lines). Bottom: Original (points) and DCT approximated curve (solid line).

the MFCC contours. Frame-based pitch and energy are generated first and are mean subtracted over the voiced part of the utterance before approximating the temporal trajectory. We use the syllable duration (number of frames) and 6 DCT coefficients per feature contour which results in 13-dimensional vectors for the prosodic and 85-dimensional vectors for the combined prosodic and MFCC contours.

## 4. Classifier

In this section, we introduce four statistical models that are used in our experimental part. The first two are flavors of Universal Background Model (UBM)-GMM models as used in speaker verification, with and without session variability compensation. The third and fourth are classical GMMs trained in generative and discriminative manner as often used in language identification. We provide most of the needed formulas to easily allow the reader to reproduce our results.

### 4.1. UBM based models

Our first two GMM systems are based on a standard Universal Background Model-Gaussian Mixture Modeling (UBM-GMM) paradigm (Reynolds et al., 2000). All GMMs used are multivariate with dimension $D$ and using diagonal co-variances.

Prior to any class-dependent model training a class-independent model is trained on the pooled feature vectors $\boldsymbol{o}$ of all development data of all classes. Following speaker recognition terminology we call this a Universal Background Model. Weights $\pi$, means $\boldsymbol{\mu}$ and variances $\boldsymbol{\Sigma}$ of the UBM are trained in a maximum-likelihood way with an Expectation-Maximization (EM) algorithm (Bishop, 2006).

EM is an iterative algorithm that alternates between estimating the responsibilities $\gamma_k(n)$ (E-Step, alignment of frame $n = 1 \ldots N$ to Gaussian components $k = 1 \ldots K$) and re-estimation of the parameters using the current responsibilities (M-Step):

E-Step:

$$\gamma_k(n) = \frac{\pi_k \mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_k \mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_k, \Sigma_k)}. \tag{4}$$

M-Step:

$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k(n) \boldsymbol{o}_n, \tag{5}$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k(n) \left(\boldsymbol{o}_n - \boldsymbol{\mu}_k^{new}\right) \left(\boldsymbol{o}_n - \boldsymbol{\mu}_k^{new}\right)^T, \tag{6}$$

$$\boldsymbol{\pi}_k^{new} = \frac{N_k}{N} \tag{7}$$

with

$$N_k = \sum_{n=1}^{N} \gamma_k(n) \tag{8}$$

and likelihood function $\mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is

$$\frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}_k|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{o}_n - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{o}_n - \boldsymbol{\mu}_k)\right\} \tag{9}$$

for feature vector $\boldsymbol{o}_n$ with feature dimension $D$.

Data log-likelihood for the whole GMM and all data $\boldsymbol{o}$

$$\ln p(\boldsymbol{o}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{10}$$

is checked for convergence after each iteration.

For UBM training we initialize a single Gaussian component with a global mean and variance of all background data and keep splitting the components in two (after several iterations when convergence of data log-likelihood is achieved) until the final size is reached. For this purpose, the copied weights $\pi$ are halved, variances $\boldsymbol{\Sigma}$ are kept and copied means $\boldsymbol{\mu}$ are shifted by $\pm 0.2\sqrt{\boldsymbol{\Sigma}}$.

Following UBM training, the individual emotion-class models are obtained by relevance Maximum-A-Posteriori (MAP) adaptation (Reynolds et al., 2000) of the mean parameters using class specific feature vectors only. Weights and variances are kept fix. The UBM mean serves as a prior for posterior distribution of class model means and the relevance factor further restricts their movement. The point estimate of the posterior mean distribution can be seen as a compromise between the prior (UBM) mean and the maximum likelihood solution (using feature vectors for emotion class $e$ only):

$$\boldsymbol{\mu}_{ek}^{MAP} = \boldsymbol{\alpha}_k \boldsymbol{\mu}_{ek}^{ML} + (1 - \boldsymbol{\alpha}_k) \boldsymbol{\mu}_k^{UBM} \tag{11}$$

with adaptation coefficients

$$\alpha_k = \frac{\sum_{n=1}^{N} \gamma_k(n)}{\sum_{n=1}^{N} \gamma_k(n) + \tau} \tag{12}$$

and relevance factor $\tau = 16$. If some components are not occupied at all by the training data, the parameters keep their prior values; while for unlimited amount of data the MAP estimate would equal the ML estimate.

During testing the models are evaluated using the log-likelihood ratio (LLR) between the class model- and the UBM log-likelihood for the test data, evaluating Eq. (10) for both the class model and UBM. For computational efficiency, only top scoring Gaussians (determined based on the UBM) are evaluated for the class models per frame. We will call this model simply *GMM-UBM* model.

The described GMM-UBM framework can be expanded to cope with intersession variability (e.g. different channel, language, gender, etc. between training and test utterances). This technique allows us to adapt the supervector of means (concatenated mean parameters of all Gaussian components) in directions of large intersession variability during verification to better match the test utterance.

In Fig. 3 we try to visualize the meaning of this technique for emotion recognition on a simple toy example. We assume GMMs containing a single mixture component each in a two dimensional feature space. The figure shows only the mean parameters of the GMMs. We should assume two utterances for each of the three emotion classes *Anger (black star), Neutral (cyan diamond)* and *Joy (magenta x-mark)*.

After the training of the UBM (blue cross) on all utterances we do one additional ML iteration using data from each utterance only. The new mean parameter ML estimates for each utterance are depicted in the figure, same colors belong to same emotion classes. It can be observed that most of the variability between different utterances belonging to the same emotion classes can be projected on a one-dimensional latent space (Intersession variability direction, dash-dotted line). This subspace can be robustly estimated on many diverse utterances belonging to different emotion classes (after UBM training, prior to class model training). Emotion class models are then derived by

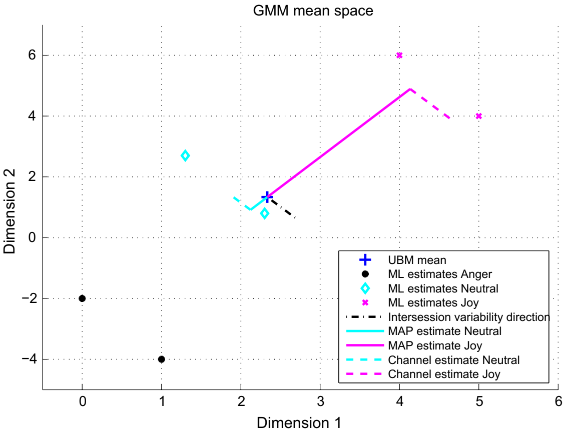


Fig. 3. Toy example of intersession variability compensation in a 2D mean parameter space. 1D subspace is estimated based on differences between utterances belonging to the same class. Model parameters can be moved along this space during verification to adapt to the test environment.

standard MAP adaptation as for the *GMM-UBM* model (shown for Neutral and Joy in plot).

During verification, the MAP adapted means of the model to be tested can be moved along the intersession variability subspace to adapt to the condition in the test utterance (acoustic condition, gender, linguistic content, etc.). This is illustrated for two utterances tested against class models for *Joy* and *Neutral* by the dashed lines drawn from the top of the solid lines (MAP estimate).

In a real application the subspace usually maps out from a very high dimensional supervector space (up to 100,000 dimensions) down to a low dimensional latent space (e.g. 50 dimensions) allowing it to robustly adapt model parameters on small amounts of data.

The adapted mean supervector can be represented as

$$\boldsymbol{m}_e + \boldsymbol{U}\boldsymbol{x}_n \tag{13}$$

and is distributed with a mean of $\boldsymbol{m}_e$ and a co-variance of $\boldsymbol{U}\boldsymbol{U}^T$. $\boldsymbol{m}_e$ is the class (emotion) dependent supervector of MAP adapted means (from standard *GMM-UBM* model). $\boldsymbol{U}$ defines the low-dimensional subspace matrix (size $DK \times S$ with subspace size $S \ll DK$) of the full GMM space with high intersession variability. The utterance dependent factors $\boldsymbol{x}_n$ define the shift of the model parameters within the subspace. These factors are assumed to be normally distributed random variables making the whole thing a probabilistic model.

The subspace is usually estimated for on a large amount of data (similar to UBM), either using Principle Component Analysis (PCA) (Burget et al., 2007) or by an EM algorithm (Kenny et al., 2008). Please refer to these citations for detailed descriptions.

Once the subspace is estimated, emotion models (or UBM) can be adapted by shifting its mean supervector in the directions given by an intersession variability subspace to better fit the test utterance data. Mathematically, this can be expressed as finding the factors $\boldsymbol{x}_r$, that maximize the following MAP criterion:

$$p(\boldsymbol{o}_r|\boldsymbol{m}_e + \boldsymbol{U}\boldsymbol{x}_r)\mathcal{N}(\boldsymbol{x}_r; \boldsymbol{0}, \mathbf{I}), \tag{14}$$

where $p(\boldsymbol{o}_r|\boldsymbol{m}_e + \boldsymbol{U}\boldsymbol{x}_r)$ is the likelihood of the test conversation $r$ given the adapted supervector (model) and $\mathcal{N}(\cdot; \boldsymbol{0}, \mathbf{I})$ denotes a normally distributed vector. Assuming a fixed occupation of Gaussian mixture components (responsibilities) by test conversation frames, $\boldsymbol{o}_n$, $n = 1, \ldots, N$, it can be shown (Brümmer, 2004) that $\boldsymbol{x}_r$ maximizing criterion (14) is given by:

$$\boldsymbol{x}_r = \boldsymbol{A}^{-1} \sum_{k=1}^{K} \boldsymbol{U}_k^T \sum_{n=1}^{N_r} \gamma_k(n) \frac{\boldsymbol{o}_n - \boldsymbol{\mu}_k}{\sigma_k}, \tag{15}$$

where $\boldsymbol{U}_k$ is the $D \times S$ part of matrix $\boldsymbol{U}$ corresponding to $k$th mixture component; $\gamma_k(n)$ is the probability of occupation mixture component $k$ at time $n$, $\boldsymbol{\mu}_k$ and $\boldsymbol{\sigma}_k$ are the mixture component's mean and standard deviation vectors and

$$\boldsymbol{A} = \boldsymbol{I} + \sum_{k=1}^{K} \boldsymbol{U}_k^T \boldsymbol{U}_k \sum_{n=1}^{N_r} \gamma_k(n). \tag{16}$$

In our implementation, occupation probabilities, $\gamma_k(n)$, are computed using UBM and assumed to be fixed for a given test conversation. This allows one to pre-compute matrix $\boldsymbol{A}^{-1}$ only once for each test conversation.

Note, that both model and UBM means are adapted to the test utterance and afterwards scoring is done exactly as for the *UBM-GMM* model (LLR).

We will call this model incorporating intersession variability compensation *ISV* model.

### 4.2. Generative and discriminative GMMs

Emotion recognition is a closed-set identification task (similar to Language identification) and usually large amounts of data are available to train the separate class models. In this section we propose to train each class model using an EM algorithm as described in the previous section for the UBM. Our assumption is that we have enough data to robustly estimate weight, mean and variance parameters for each emotion class individually.

Furthermore, we propose to re-estimate the model parameters using a discriminative training technique successfully applied to language identification (Matejka et al., 2006).

As depicted in Fig. 4 discriminative techniques aim to precisely model the boundary between the competing models in such a way that the correct estimation of class affiliation is improved rather than maximizing the likelihood of the training data. This way model parameters are mostly used to estimate precisely the boundaries between separable regions in the features space. Highly overlapping areas are neglected.

Our first set of models is trained per class under the conventional Maximum Likelihood (ML) framework, as used for the UBM (see Section 4.1, Eqs. (4)–(10)), but only using class specific data. Note, that we re-estimate not only means, but also weights and variances per emotion class. We will call these models simply *ML* models.

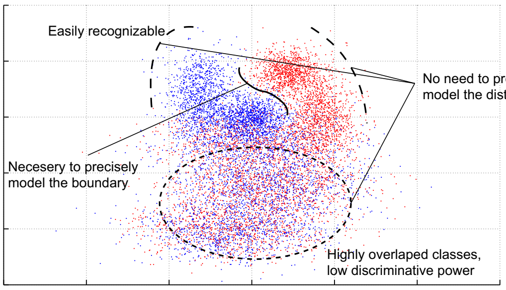


Fig. 4. Effect of discriminative training for two classes in 2D feature space. The model parameters are used to precisely model the boundary between separable data while highly overlapping areas are neglected.

These serve as a starting point for further discriminative re-estimations of means and variances using the Maximum Mutual Information (MMI) criterion.

Unlike in the case of ML training, which aims to maximize the overall likelihood of training data given the transcriptions, the MMI objective is to maximize the posterior probability of correctly recognizing all training segments (utterances):

$$\mathcal{F}_{MMI} = \sum_{r=1}^{R} \ln \frac{p(\boldsymbol{o}_r|\boldsymbol{\mu}_{e^+}, \boldsymbol{\Sigma}_{e^+}, \boldsymbol{\pi}_{e^+})}{\sum_{e=1}^{E} p(\boldsymbol{o}_r|\boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e, \boldsymbol{\pi}_e)}. \tag{17}$$

where the numerator is the likelihood of $r$-th training segment $\boldsymbol{o}_r$; given the correct emotion class model of the segment, $e^+$; $R$ is the number of training segments and the denominator represents the overall probability density, $p(\boldsymbol{o}_r)$ (likelihood given any emotion class). So, the MMI parameter re-estimates aim to maximize the ration between true class likelihood and overall likelihood of each segment.

It can be shown (Povey, 2003) that the MMI objective function (17) is increased by re-estimating model parameters using extended Baum-Welch algorithm (similar to standard EM training) with the following formula for updating mean and variances:

$$\mu_{ek}^{new} = \frac{\theta_{ek}^{num}(\boldsymbol{o}) - \theta_{ek}^{den}(\boldsymbol{o}) + 2\gamma_{ek}^{den}\mu}{\gamma_{ek}^{num} + \gamma_{ek}^{den}}, \tag{18}$$

$$\Sigma_{ek}^{new} = \frac{\theta_{ek}^{num}(\boldsymbol{o}^2) - \theta_{ek}^{den}(\boldsymbol{o}^2) + 2\gamma_{ek}^{den}\left(\Sigma_{ek} + \mu_{ek}^2\right)}{\gamma_{ek}^{num} + \gamma_{ek}^{den}} - \mu_{ek}^{new^2}. \tag{19}$$

The terms:

$$\theta_{ek}^{num}(\boldsymbol{o}) = \sum_{r=1}^{R}\sum_{n=1}^{N_r} \gamma_{ekr}^{num}(n)\boldsymbol{o}_r(n), \tag{20}$$

$$\theta_{ek}^{num}(\boldsymbol{o}^2) = \sum_{r=1}^{R}\sum_{n=1}^{N_r} \gamma_{ekr}^{num}(n)\boldsymbol{o}_r(n)^2,$$

$$\gamma_{ek}^{num}(\boldsymbol{o}) = \sum_{r=1}^{R}\sum_{n=1}^{N_r} \gamma_{ekr}^{num}(n),$$

are mixture component specific first and second order statistics and occupation counts corresponding to the numerator of the objective function (17). Denominator statistics can be expressed by similar equations, where all superscripts *num* are merely replaced by *den*. Note that the numerator statistic are ordinary ML statistics. Therefore, the numerator posterior probability of occupying mixture component *ek* by $n$-th frame of training segment $r$,

$$\gamma_{ekr}^{num}(n) = \begin{cases} \gamma_{ekr}(n) & \text{for } e = e^+, \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

is non-zero only for mixture components corresponding to the correct emotion class. To estimate the posterior probabilities for the denominator:

$$\gamma_{ekr}^{den}(n) = \gamma_{ekr}(n)\frac{p(\boldsymbol{o}_r|\boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e, \boldsymbol{\pi}_e)}{\sum_{q=1}^{E} p(\boldsymbol{o}_r|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q, \boldsymbol{\pi}_q)}. \tag{22}$$

Note, that the fraction on the right-hand side is the posterior probability of the current emotion class given the whole segment that $n$ belongs to.

Finally,

$$\gamma_{ekr}(n) = \frac{\pi_{ek}\mathcal{N}(\boldsymbol{o}_r(n)|\mu_{ek}, \Sigma_{ek})}{\sum_{j=1}^{K} \pi_{ej}\mathcal{N}(\boldsymbol{o}_r(n)|\mu_{ej}, \Sigma_{ej})} \tag{23}$$

where $\pi_{ek}$ is mixture component weight and $K$ is the number of mixture components in model $e$.

Starting from the ML models of final size, the mean and variance parameters are re-estimated using MMI for several iterations.

For both models, verification is done frame-by-frame for the test utterance with full log-likelihood computation according to Eq. (10). Note, that we always evaluate all Gaussian components for these two model types.

## 5. Experiments on the FAU Aibo emotion corpus

In this section we present experimental results to evaluate the techniques presented in Sections 2–4. All used feature configurations and classifiers are summarized in Table 1. Experiments on feature types and modeling techniques are performed on the FAU Aibo corpus.

### 5.1. Database

The FAU AIBO database is a corpus with recordings of children of age 10 to 13 interacting with a pet robot called Aibo. The emotionally colored speech is non-rehearsed, as the children believed that the robot was following their commands, so their reactions evoke emotions due to behavior or misbehavior. Actually, the actions of the robot were in a fixed order, controlled by an operator and similar for all participants.

The whole corpus consists of 9.2 hours of high quality speech which was annotated by human labelers and assigned to emotional classes by majority voting. All sessions are split on a chunk level to achieve homogeneity of emotional state within a unit and results in about 18,000 chunks.

The database was recorded at two different schools, consisting in a total number of recordings of 51 children. The first portion consists of 13 male and 13 female speakers. Within the Emotion Challenge 2009, the first part was provided as a combined training and development set, while the second part was defined to be the test set. The emotion labels for the second part were not provided and results could only be evaluated within the Interspeech 2009 Emotion Challenge. As a consequence, we will provide two different results in this chapter. First, we will describe progress in system development on our own defined development set and afterwards, we will give the official results obtained in the challenge with our final systems.

All annotated emotion labels of each chunk were mapped to two broader sets of emotions: A 5-class set

Table 1
Summary of feature sets and model types used in experimental part.

| Feature type | Description | Dimension |
|---|---|---|
| MFCC | C0+12 MFCCs with CMS, VAD | 13 |
| RASTA | C0+12 MFCCs with RASTA and CMS, VAD | 13 |
| RASTA-Δ | C0+12 MFCCs with RASTA and deltas, CMS, VAD | 26 |
| RASTA-ΔΔ | C0+12 MFCCs with RASTA, deltas and double deltas, CMS, VAD | 39 |
| RASTA-ΔΔΔ | C0+12 MFCCs with RASTA, deltas, double and triple deltas, CMS, VAD | 52 |
| SDC | C0+6 MFCCs+delta cepstra over 21 frames, CMS, VAD | 56 |
| DPE | Duration+syllable contours (6 DCT coefficients each) for pitch and energy | 13 |
| DPEC | Duration+syllable contours (6 DCT coefficients each) for pitch, energy and MFCCs | 85 |
| **Model type** | **Description** | **Components** |
| GMM-UBM | GMM with MAP adapted means from UBM | 8–128 |
| ISV | GMM with MAP adapted means from UBM and intersession variability compensation | 8–128 |
| ML | ML-trained GMM (weights, means, variances) | 16–128 |
| MMI | ML-trained GMM (weights, means, variances) with further MMI training (means, variances) | 16–128 |

containing **A**nger, **E**mphatic, **N**eutral, **P**ositive and **R**est, and a 2-class set comprising **NEG**ative and **IDL**e. Detailed information on the database and its design is given in (Steidl, 2009). To keep our experimental part clear for the reader we present only results on the 5-class task (obviously the more difficult task).

The total number of chunks available for training/development of the 5-class models are in Table 2. Note that the numbers differ from Schuller et al. (2009), as our voice activity detection did not identify any speech frames for several chunks.

### 5.2. Development set

We use subsets of the training data for system development. We use a full jackknifing approach for the whole training set. Thirteen splits are created out of the training set, each excluding 1 male and 1 female (so speaker in training and test are always distinct), resulting in circa 700 chunks for the testing of each split. We train a separate system for each split on the remaining chunks. This is a very expensive procedure, but this way we can use all available data for training and testing, while the training and test portions are always distinct. Results are presented in terms of two accuracies: The *Weighted Accuracy (WA)* means the percentage of correctly recognized chunks, in the total for all chunks over all classes of the development data. The *Unweighted Accuracy (UA)* means the percentage of correctly recognized chunks per class, which are then averaged over all classes. As the class affiliation is highly unbalanced (see Table 2), we will use the unweighted accuracy as our primary measure for system development.

Table 2
Number of chunks in the AIBO corpus development set to train each classifier for 5 classes.

| Anger | Emphatic | Neutral | Positive | Rest | ∑ |
|---|---|---|---|---|---|
| 830 | 1890 | 5024 | 616 | 642 | 9002 |

Table 3
Results for static MFCCs features with longer temporal context using *GMM-UBM* with 64 components [%].

| Static | | | Longer context | | |
|---|---|---|---|---|---|
| Feature | UA | WA | Feature | UA | WA |
| MFCC | 36.4 | 40.4 | RASTA-Δ | 41.8 | 41.3 |
| RASTA | **37.4** | 40.9 | RASTA-ΔΔ | **43.5** | 42.9 |
| | | | RASTA-ΔΔΔ | 42.6 | 40.7 |
| | | | SDC | 41.9 | 41.0 |

### 5.3. Spectral features

We start with investigations of spectral features using a fixed classifier to compare the performance of the different feature sets. We use a *GMM-UBM* system for this purpose. Preliminary experiments indicate that 64 Gaussians work well for the first *GMM-UBM* system.

As we are using an adaptation from the background to class model it is important to define a balanced set for the UBM training due to the unbalanced amount of class affiliation in the training data (see Table 2). Otherwise, the background model would be biased to the more dominant classes (Neutral and Emphatic) and adapted models for the under-represented classes might be poor. For this purpose, we select 500 chunks from each of the 5 classes to train a model that serves as the UBM. Emotion class models are then obtained by relevance MAP adaptation of the mean parameters.

Results are presented in the left column of Table 3. With 36.4%, the unweighted accuracy is very low for the simple MFCC features. Still, these results correspond with the results reported in a similar test set of the AIBO corpus for a frame based HMM system (Schuller et al., 2009). A significant[1] improvement is achieved through the use of a simple RASTA filter.

The use of Vocal Tract Length Normalization did not give conclusive results and no significant gains could be

[1] At a significance level of $\alpha = 0.1$.

Table 4
Results for syllable based feature contours modeled by 64 component *GMM-UBM* [%].

| Feature | UA | WA |
|---|---|---|
| DPE | 32.3 | 39.6 |
| DPEC | **36.0** | 38.3 |

achieved. The ineffectiveness of VTLN might be explained by an analysis of the observed distribution of the warping factors. Generally, for data of adults a separation of males and females in the form of a bimodal distribution can be observed. In our experiments there was no separation of warping factors for male and female speech at all which might be caused by the fact that we handle children's speech. Probably a more adjusted grid search than using warping factors 0.88–1.12 (which is somehow optimized for adult speech) can be more effective. As a consequence, we use only RASTA processed features for the following experiments.

As a next step we apply techniques to cover a broader temporal context. Up to now features only model a quasi-static period of approximately 30 ms. We augment the RASTA features with their delta (RASTA-Δ), double-delta (RASTA ΔΔ) and triple-delta (RASTA-ΔΔΔ) regression coefficients.

Results are presented in the column on the right of Table 3. Significant improvements are obtained through all examined configurations and the task clearly benefits from broadening the temporal context. The best results are achieved with RASTA-ΔΔ features which significantly outperform the single delta and SDC features.

One interpretation might be that enlarging the context keeps improving the accuracy but triple delta and SDC feature dimensions are already too high for this scenario. Keep in mind that a higher feature dimension also raises the free parameters in the model dramatically.

Following these experiments, we will use the RASTA-ΔΔ coefficients as our primary spectral feature set.

### 5.4. Prosodic features

The following experiments are performed to evaluate the prosodic features proposed in Section 3.

We use the same *GMM-UBM* model type as for our previous experiments with features containing the following feature subsets: duration and temporal contours of pitch and energy (DPE) and duration, pitch, energy and MFCC temporal contours (DPEC, see also Table 1).

Results are presented in Table 4. The best results of 36% UA are achieved with the DPEC features. These features show a similar performance as the simple MFCC features without any temporal context. However, the spectral frame-based features incorporating an equal temporal context still perform significantly better. This is a result we also observe in speaker or language identification. High-level features like these usually perform worse on their own but add complementary information. This is then exploited by score-level fusion of the diverse recognition systems.

Another reason for the huge degradation might be the fact that we use statistical classifiers with very little data. As these features are based on syllable regions spanning a context of up to several hundred milliseconds, often only a few or no feature vectors can be extracted per utterance. Clearly, the performance of this feature type suffers greatly from the fact that the test utterances are very short in the AIBO corpus.

### 5.5. GMM-UBM models

Now we start evaluating the modeling techniques proposed in Section 4. For this purpose, we will use the spectral RASTA-ΔΔ features that performed best in the previous section.

After selecting 64 Gaussians somehow ad-hoc for the initial feature experiments, additional experiments are carried out to find optimal sizes for *GMM-UBM* as well as for *ML* systems for this task.

The use of up to 2048 Gaussian components is typical in high-performing speaker and language identification systems, where much more data is available for each class or for the background model (Burget et al., 2007; Matejka et al., 2006). The used databases of emotional speech are rather small, so (1) we have little data to train the background model and the class model; and (2) the test utterances are also quite short (only up to several seconds). For this reason, we expect the optimum GMM size to be much smaller than for SID/LID systems.

As we use an EM training algorithm that splits Gaussian components after some iterations, we evaluate GMM sizes from 8 to 128, doubling the size after each step. It should be noted that we will also provide class-specific accuracies in this section to show the relation of the GMM size and the amount of available training data.

Results in Table 5 for a GMM-UBM system indicate that a size of 64–128 components is optimal for this task. Using a larger number of mixture components did not increase UA. WA usually kept rising as the major classes (like Neutral) benefit from larger amount of model parameters while the others get overtrained.

### 5.6. ML models

Furthermore, for the proposed model types in Section 4.2 an independent GMM is trained for each class on the

Table 5
Unweighted, weighted and class specific accuracies for different GMM sizes with RASTA-ΔΔ features for *GMM-UBM*.

| GMM size | UA | WA | A | E | N | P | R |
|---|---|---|---|---|---|---|---|
| 8 | 40.8 | 37.8 | 62.3 | 33.5 | 35.9 | 69.0 | 3.1 |
| 16 | 41.7 | 41.3 | 62.1 | 33.3 | 42.5 | 67.1 | 3.6 |
| 32 | 42.4 | 43.6 | 59.6 | 37.1 | 45.8 | 66.1 | 3.4 |
| 64 | 43.5 | 42.9 | 60.8 | 36.5 | 43.6 | 71.4 | 5.3 |
| 128 | **43.6** | 43.7 | 61.2 | 35.8 | 45.3 | 70.8 | 5.0 |

available target training data only, without any adaptation from a background model. According to Table 2 we only have several hundred chunks available for some classes. For this purpose and to further confirm the GMM size, we proceed with experiments using a different number of Gaussian components for simple *ML* trained models without any background model adaptation.

We evaluate sizes of 16 to 128 Gaussians. The results are presented in Table 6.

Interestingly, for these models we see a similar trend as for the MAP adapted models. We obtain the best results of about 44% UA with 32 and 64 Gaussians. Again, 64 Gaussians seem to be a good choice. Also the models for which only small amount of data is available, such as **A**, **P** or **R**, already seem to get overtrained with 128 Gaussians.

If we compare the results for the *GMM-UBM* system and the *ML* system in Tables 5 and 6 we observe a similar overall performance. Comparing same sized models, we see that the *ML* models are significantly better for the smaller models. This seems reasonable as the small *ML* models might have more discriminative power due to their individual weight and variance parameters. However, for the larger models the amount of training data might still be too small to estimate all these parameters robustly.

### 5.7. *MMI models*

After evaluating the two basic GMM models we move on with experiments using more sophisticated modeling approaches.

First, we use the MMI criterion to retrain all generative class GMMs (*ML* models) to discriminative models. This is done in addition to 10 iterations, always increasing the MMI objective function in (17). Comparing the numbers in Table 7 for *MMI* models with previous *ML* experiments (see Table 6) gives somewhat disappointing results.

Except for the small GMM with 16 Gaussians (not significant), all other recognition rates even decrease due to MMI training. This loss of performance is also not significant but seems to show a trend. Only when looking at very small number of Gaussians (e.g. 2) we could spot a significant gain due to MMI, but these models obviously perform much worse than the larger ones.

It should be mentioned here that the proposed technique of discriminative re-training of models leads to huge improvements on NIST evaluation sets for language identification (Matejka et al., 2006) with similar number of classes. More than 50% improvement can be achieved on 30 s long test utterances. Interestingly, on 3 s long utterances (which is more similar to our scenario here) the gain also reduces to less than 10% relative. Another difference is the amount of data to train the class models, which is much higher (hundreds of hours per class) in the case of the NIST LID task (NIST, 2005).

Table 6
Unweighted, weighted and class specific accuracies for different GMM sizes with RASTA-ΔΔ features for the *ML* model.

| GMM size | UA | WA | A | E | N | P | R |
|---|---|---|---|---|---|---|---|
| 16 | 42.7 | 46.2 | 54.0 | 40.2 | 47.5 | 46.4 | 16.2 |
| 32 | **44.3** | 48.2 | 55.9 | 45.5 | 51.5 | 46.3 | 22.1 |
| 64 | 44.0 | 49.2 | 51.5 | 45.0 | 54.1 | 46.3 | 23.1 |
| 128 | 42.8 | 51.3 | 48.6 | 43.6 | 59.6 | 42.9 | 19.2 |

Table 7
Unweighted, weighted and class specific accuracies for different GMM sizes with RASTA-ΔΔ features for the *MMI* model.

| GMM size | UA | WA | A | E | N | P | R |
|---|---|---|---|---|---|---|---|
| 16 | 42.9 | 46.9 | 53.6 | 49.1 | 48.8 | 44.5 | 18.5 |
| 32 | **44.2** | 48.5 | 53.4 | 45.7 | 52.3 | 44.5 | 25.0 |
| 64 | 43.7 | 49.5 | 49.5 | 45.1 | 55.0 | 44.0 | 24.8 |
| 128 | 42.2 | 51.4 | 47.1 | 43.4 | 60.5 | 39.9 | 20.2 |

### 5.8. *ISV models*

In the following experiments we want to evaluate the intersession variability compensation approach as proposed in Section 4.1. The system is mainly a *GMM-UBM* system as used in the initial feature experiments with additional intersession variability compensation during testing.

As a first step the low dimensional subspace defining the directions of intersession variability has to be estimated on the training data. The usage of the available training data is crucial during this step and defines what kinds of intersession variability can be compensated for.

The AIBO database comprises many chunks for the same class and the same speaker. So we can learn differences according to acoustic environment, speaker or linguistic content. Our main assumption is that we do not have many channel effects caused by different microphones or transmission channels. As all recordings are done using the same equipment in the same room, the within-class-covariance will mainly cover speaker and intrinsic variations (Shriberg et al., 2009). Still, acoustic channel compensation might be an issue for the test set as this is recorded in a different school under different acoustic conditions.

As the segments are rather short in this database we use a method to learn more reliable subspace directions. We concatenate all segments belonging to the same speaker and class and estimate $\boldsymbol{U}$ as to describe the difference between speakers. This way our intersession variability compensation serves more as a speaker compensation than an acoustic channel compensation.

Before starting the subspace training, we initialize $\boldsymbol{U}$ by PCA (Burget et al., 2007) to ensure a good starting point and faster convergence. Then we iteratively re-train $\boldsymbol{U}$ in 10 iterations.

Once the subspace is estimated, emotion class models are trained by relevance MAP adaptation exactly as for the *GMM-UBM* models. Also, the scoring part itself (LLR) is the same. The only difference is that we adapt the obtained MAP means towards the test utterance along the low-dimensional subspace $\boldsymbol{U}$. This is done by estimating the "channel" factors $\boldsymbol{x}$ for each test utterance using Eqs. (15), (16).

We perform several experiments to determine the optimal number $S$ of intersession variability directions (size of the subspace). Fig. 5 shows unweighted accuracies for up to 5 subspace directions. We can observe that using more than 1 eigenchannel always decreases the performance. We get non-significant improvement over the relevance MAP model with 44.2% for 1 eigenchannel (dashed line), but it drops consistently when increasing the number of subspace directions, which significantly decreases the performance.

One explanation might be that the test utterances in this corpus are simply to short (often below one second of speech) to reliably estimate the $\boldsymbol{x}$ factors that control the adaptation of the model mean parameters. Similar degradation of intersession compensation techniques due to small amount of test data has been observed for speaker (Dehak et al., 2009) as well as language identification (Hubeika et al., 2008) tasks incorporating only a few seconds of speech. Also, the subspace $\boldsymbol{U}$ is usually trained on hundreds of hours of speech.

### 5.9. System calibration/Fusion

It is advisable to calibrate the system outputs as the obtained scores for our systems do not represent proper posterior probabilities for the classes. A certain GMM may generally produce higher scores than the others in the set. Furthermore, a consequent step is to fuse several of the systems that incorporate partly complementary information, as we have created many different systems based on diverse features and modeling techniques. We have observed huge gains in performance using this technique (Brümmer et al., 2007) even for system configurations that differ only slightly (e.g. only different feature sets).

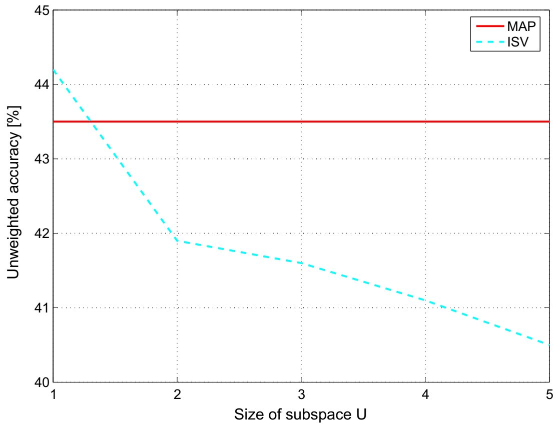


Fig. 5. Effect of eigenchannel subspace size on AIBO corpus. UA for RASTA-ΔΔ features and *ISV* model with 64 components.

For these purposes we use multi-class linear logistic regression (MLLR) (Brümmer and du Preez, 2006) to perform calibrated fusion of our system outputs. Posterior probabilities of class $\mathcal{C}_e$ given the score vector $\phi$ are then given by:

$$p(\mathcal{C}_e|\phi) = \frac{\exp(a_e)}{\sum_{j=1}^{E}\exp(a_j)} \tag{24}$$

with activations

$$a_e = \mathbf{w}_e^T\phi \tag{25}$$

and $\phi$ containing the concatenated scores from all systems to be fused. The fusion parameters $\mathbf{w}_e$ are trained on each split of the development set and are then averaged to ensure fair circumstances.

First we perform fusions of two systems that are using the same features but four different modeling techniques. Fusion results for all combinations are presented in Table 8 and are mostly better than the best single ISV system with 44.2%. Significant gains are achieved due to fusion of two heterogeneous systems, like one background model based (*GMM-UBM* or *ISV* system) and one standard GMM model (*ML* or *MMI* system). Fusion of systems where one is derived from the other, like *ML* and *MMI*, results only in a small improvement. Fusion of all 4 systems does not result in further improvement.

Furthermore, we evaluate the effect of fusing systems using different feature sets while keeping the modeling approach fixed (*ISV*). For this purpose we have selected 4 different feature sets that should be most complementary. We select the RASTA MFCCs without further temporal context (37.4%); the SDC features (41.9%); the simple prosodic DPE features (32%); and our standard RASTA-ΔΔ features (44.2%). Results in Table 9 show the same trend as our previous fusion experiments. All combinations are better than the best incorporated single system. Significant gains can be achieved and the best result of 45.9% is obtained for a fusion of RASTA-ΔΔ and SDC features. Again, we fuse all 4 systems without any further improvement.

To conclude these experiments we change both variables (features and modeling techniques) at once. We fuse different combinations but without any further improvement.

### 5.10. Emotion challenge 2009

This section shows the results for the systems we have selected to submit for the official Open Performance

Table 8
Results (UA) for fusion of 2 systems with same features (RASTA-ΔΔ) and different modeling approaches [%].

| | GMM-UBM | ISV | ML | MMI |
|---|---|---|---|---|
| GMM-UBM | – | 44.1 | **45.5** | 45.3 |
| ISV | | – | **45.5** | 45.1 |
| ML | | | – | 44.3 |

Table 9
Results (UA) for fusion of 2 systems with same modeling technique (*ISV*) but different feature sets [%].

| | SDC | RASTA-ΔΔ | DPE |
|---|---|---|---|
| RASTA | 43.9 | 44.5 | 39.3 |
| SDC | – | **45.9** | 44.5 |
| RASTA-ΔΔ | | – | 44.2 |

Sub-Challenge (Schuller et al., 2009) of the Interspeech Emotion Challenge 2009. Results are presented with the official metric on 5-class tasks, similar to our results on the development set. As classes are highly unbalanced, the rules stipulated the use of the unweighted average recall (UA) as the primary measure and the weighted average recall (WA) as the secondary measure.

We have selected the four different modeling approaches (*ML, MMI, GMM-UBM, ISV*) we used in the system development for the best performing features on the test set. They are based on MFCCs generated with RASTA filter, double-deltas, CMS and VAD (RASTA-ΔΔ). Note, that the scores computed on the test set could only be uploaded up to 25 times. So we had to select the most promising configurations. In Schuller et al. (2009) baseline recognition results on the test set are provided for two different baseline systems. A dynamic modeling approach using frame-based features and a Hidden-Markov-Model (HMM) as a classifier; and the second static approach uses high-dimensional chunk based features fed to a Support Vector Machine classifier. The best baseline results on the proposed primary measure are 35.9% for the HMM baseline and 38.2% for the SVM baseline.

Table 10 shows the results for the 5-class task for the 4 submitted models. We achieve the best results for the *ISV* system with 41.3%. Surprisingly, the *ML* and the *MMI* system perform significantly worse with only about 38.5%, unlike than on the development set. This might indicate that even the ML trained model is already over-adapted to the training data and does not generalize well. The simple GMM-UBM system performs significantly better than the ML/MMI approaches. We get improvement (not significant) from the intersession variability compensation. On the UA we achieve a 15%/8% relative improvement to the HMM and SVM modeling, respectively, which was provided as a baseline.

As proposed in the last section, we want to combine several complementary systems to achieve the best results. We select the most promising fusion of two systems as evaluated in Table 9. We fuse 2 systems using the same ISV model with 64 Gaussians, one with RASTA-ΔΔ features and one with SDC features. The fusion parameters are the same as used in our system development.

Table 10
Submitted systems for the 5-class task [%]. All using RASTA-ΔΔ features.

| Feature | UA | WA |
|---|---|---|
| GMM-UBM | 40.8 | 41.0 |
| JFA | **41.3** | 43.9 |
| ML | 38.5 | 45.4 |
| MMI | 38.7 | 46.0 |

We get another improvement and achieve an unweighted average recall of 41.7%. This is the highest recognition rate achieved in the Interspeech 2009 Emotion Challenge for the 5-class task. Still, our result was not significantly better than that of some other participants. The organizers (Schuller et al., 2009) could show that further fusion of the (completely independent) participating systems could significantly increase the recognition rate to over 44%.

## 6. Berlin database of emotional speech

In this section we will present some additional experiments mainly to further investigate the effect of intersession compensation for emotion recognition. As test utterances are extremely short on the FAU Aibo corpus we selected a database with longer test utterances. The Berlin Database of Emotional Speech (Burkhardt et al., 2005) consists entirely of whole sentences that are several seconds long.

### 6.1. Database

This database contains acted emotional speech. Ten actors (5 male and 5 female) simulated seven different emotions on ten German utterances (5 short and 5 long). Emotion classes are **A**nger, **F**ear, **N**eutral, **J**oy, **S**adness, **D**isgust and **B**oredom. The recordings are studio-quality and the whole database contains 535 sentences. It should be noted, that although the single utterances are longer than for the AIBO corpus, the overall amount of speech data is much smaller (less than one hour).

### 6.2. Development set

Similar to the AIBO database we use a full jackknifing approach for the whole training set. Ten splits are created out of the training set, each excluding one speaker. The actual number of sentences available to train the classifiers are depicted in Table 11. Similar to Section 5.2, results are presented in terms of unweighted accuracy (UA). It should be noted, as the amount of speech data for class **D** is extremely low and preliminary testing fails completely in this class, we discard class **D** from our development set and take only 6-classes into account.

### 6.3. ISV model

We perform experiments on a similar system as used for the Interspeech Emotion Challenge. We create MFCC

Table 11
Number of utterances in the Berlin Database of Emotional Speech to train each classifier.

| A | B | D | F | J | S | N | $\sum$ |
|---|---|---|---|---|---|---|---|
| 127 | 81 | 46 | 69 | 71 | 62 | 79 | 535 |

features, apply the RASTA filter and CMS and augment the features with delta and double-deltas. Afterwards, speech frames are selected using voice activity detection (RASTA-ΔΔ).

These spectral features are first used to train a UBM with 64 Gaussians. We again use a class-balanced data set for background model training. A single UBM for each split consists of approximately 300 sentences, 50 for each class. After UBM training we train the intersession variability subspace $\boldsymbol{U}$. We use the same recipe for subspace estimation as in the previous experiments: PCA initialization of $\boldsymbol{U}$ with successive ML-training. We again concatenate all utterances per speaker to train the intersession variability subspace. The whole database was recorded in an anechoic chamber using high-quality equipment so channel effects are minimal. Effects of speaker normalization might be even more meaningful than for the AIBO corpus as the database consists of adult speech.

Experiments are carried out to investigate the effect of intersession compensation for this database. For this purpose we train and evaluate *GMM-UBM* and *ISV* models as described in the previous sections. In Fig. 6 an interesting trend can be observed which is different from the experiments on the AIBO corpus. While we reach an unweighted accuracy of 57% using relevance MAP, we achieve a significant improvement by using the same system incorporating intersession variability compensation. As depicted by the dashed line in Fig. 6 we reach a recognition rate of 63% with the use of one subspace direction. The use of a larger subspace further increases the performance and the best unweighted accuracy of 67% is achieved with a subspace size of 5. This is a significant improvement of an absolute 10% UA over the *GMM-UBM* baseline.

We are aware that better recognition rates have been reported on this database. In (Schuller et al., 2006) accuracies of over 80% are reached but only by using much more complex large-scale feature sets. For these studio-quality recordings, features like pitch and voice quality will be of high accuracy and might explain the huge difference in recognition performance. In (Gaurav, 2008), frame-based MFCCs using GMMs are also evaluated and performed in a similar way to our baseline system. Furthermore, GMMs are outperformed by SVM approaches in that work. Our conclusion for the performance gap to the state-of-the art SVM systems is that SVMs might be better suitable to handle the general small amount of training data in this database.

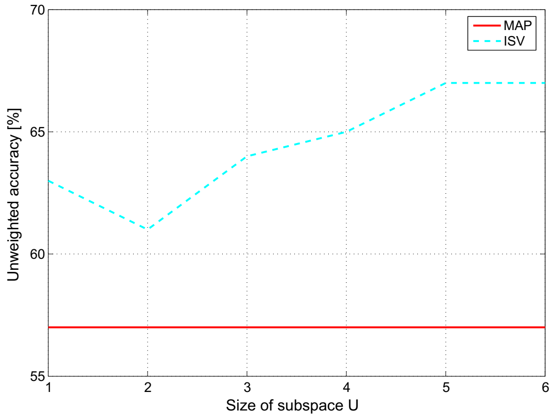


Fig. 6. Effect of eigenchannel subspace size on Berlin Database of Emotional Speech corpus. UA for RASTA-ΔΔ features and *ISV* model with 64 components.

Nevertheless, our experiments show the capability of intersession compensation techniques for emotion recognition.

## 7. Conclusions

We show that feature extraction and statistical modeling methods that are usually used in speaker and language recognition can be successfully used for emotion recognition as well.

We could achieve the best results for the 5-class task in the Interspeech Emotion Challenge 2009 and significantly outperformed the provided state-of-the-art baseline systems.

The submitted system incorporated quite simple acoustic features. We did not make use of excessive spectral, prosodic or lexical features. Eventually, we used two different feature sets both derivatives of MFCC features. Several experiments on our development set indicated that MFCC features using RASTA filter and augmented with first and second order derivatives performed the best for this task. It should be noted, that this feature set is very close to those used in automatic speech recognition. As a complementary feature set we use Shifted Delta Cepstra with an even broader temporal context.

Our prosodic feature set showed bad performance compared to the spectral features. While this is a common effect also observed in other fields of speech based pattern recognition tasks, we can conclude that in this case the given test utterances are really too short to exploit a syllable based long-temporal span feature extraction. Future work should consider exploiting a simpler prosodic feature set like frame based pitch values or functionals computed on shorter fixed size windows.

The proposed GMM based modeling approaches generally perform very well. However, the more sophisticated approaches, namely discriminative training and intersession variability compensation, were not convincing on the FAU AIBO corpus. While both approaches have proven their potential in terms of language identification we could only reach marginal improvements. Our conclusion is that this effect is mainly due to the short test utterances and the general small amount of training data per class. In the mentioned NIST evaluations for language identification the core condition consists of test utterances with durations of 30 s. In this task MMI as well as intersession variability compensation has shown up to 50% relative improvement, while on a 3 s task the gain degrades to approximately 10%

relative improvement, both for MMI and intersession variability compensation.

That there is indeed a capability for intersession variability compensation for emotion recognition is shown in the Berlin Database of Emotional Speech. Here we can obtain significant gains through the use of the *ISV* model. Still, it should be mentioned that in both cases we used *ISV* mainly to reduce the effects of intersession variability representing speaker characteristics instead of channel characteristics as is usually done.

Large-scale feature SVM modeling still seems to be superior on acted non-spontaneous studio-quality recordings, unlike that on real-world data. Our impression is that prosodic and voice-quality features are very accurate on this type of recordings and yield the high accuracies. Still, SVMs seem to be a good choice to handle very small amounts of training data while generative statistical models like GMMs get simply overtrained.

Furthermore, we could show that system combinations by score level fusion can significantly enhance performance. In conclusion, in this way diverse modeling techniques (like SVM or GMMs) and feature sets (acoustic, prosodic, chunk or frame based, etc.) can be exploited for high accuracy in emotion recognition tasks.

## References

Batliner, A., Steidl, S., Schuller, B., Seppi, D., Laskowski, K., Vogt, T., Devillers, L., Vidrascu, L., Amir, N., Kessous, L., 2006. Combining efforts for improving automatic classification of emotional user states. In: Proceedings of IS-LTC, pp. 240–245.

Bishop, C., 2006. Pattern recognition and machine learning.

Brümmer, N., 2004. Spescom DataVoice NIST 2004 system description. In: Proceedings NIST Speaker Recognition Evaluation 2004, Toledo, Spain, June. 2004.

Brümmer, N., Burget, L., Cernocky, J., Glembek, O., Grezl, F., Karafiat, M., van Leeuwen, D.A., Matejka, P., Schwarz, P., Strasheim, A., 2007. Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006. IEEE Trans. Audio, Speech Lang. Process. 15 (7), 2072–2084.

Brümmer, N., du Preez, J., 2006. Application-independent evaluation of speaker detection. Comput. Speech Lang. 20 (2–3), 230–275.

Burget, L., Matejka, P., Schwarz, P., Glembek, O., Cernocky, J., 2007. Analysis of feature extraction and channel compensation in a GMM speaker recognition system. IEEE Trans. Audio, Speech, Lang. Process. 15 (7), 1979–1986.

Burkhardt, F., Paeschke, A., Rolfes, M., Sendlmeier, W., Weiss, B., 2005. A database of german emotional speech. In: Ninth European Conference on Speech Communication and Technology.

Cohen, J., Kamm, T., Andreou, A., 1995. Vocal tract normalization in speech recognition: Compensating for systematic speaker variability. J. Acoust. Soc. Amer. 97, 3246.

Davis, S., Mermelstein, P., 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans. Audio, Speech Lang Process. 28 (pp. 1–4), 357–366.

Dehak, N., Kenny, P., eda Dehak, R., Dumouchel, P., Ouellet, P., 2009. Front-end factor analysis for speaker verification. IEEE Trans. Audio, Speech Lang. Process., 1–23.

Gaurav, M., 2008. Performance analysis of spectral and prosodic features and their fusion for emotion recognition in speech. In: Spoken Language Technology Workshop, 2008. SLT 2008. IEEE, pp. 313–316.

Hermansky, H., Morgan, N., 1994. Rasta processing of speech. IEEE Trans. Speech Audio Process. 2 (pp. 1–4), 578–589.

Hubeika, V., Burget, L., Matejka, P., Schwarz, P., 2008. Discriminative training and channel compensation for acoustic language recognition. In: Proceedings of Interspeech, 1990–9772.

Kenny, P., Ouellet, P., Dehak, N., Gupta, V., Dumouchel, P., 2008. A study of inter-speaker variability in speaker verification. IEEE Trans. Audio, Speech, Lang. Process. 16 (5), 980–988.

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

Kockmann, M., Burget, L., 2008. Contour modeling of prosodic and acoustic features for speaker recognition. In: Spoken Language Technology Workshop, SLT 2008. IEEE, pp. 45–48.

Kockmann, M., Burget, L., Cernocky, J., 2009. Brno University of technology system for interspeech 2009 emotion challenge. In: Proceedings of Interspeech, Brighton, pp. 348–351.

Matejka, P., Burget, L., Glembek, O., Schwarz, P., Hubeika, V., Fapso, M., Mikolov, T., Plchot, O., Cernocky, J., 2008. But language recognition system for NIST 2007 evaluations. In: Proceedings of Interspeech.

Matejka, P., Burget, L., Schwarz, P., Cernocky, J., 2006. Brno University of Technology system for NIST 2005 language recognition evaluation. In: Proceedings of Odyssey.

NIST, 2005. The 2005 NIST language recognition evaluation plan, pp. 1–6.

Povey, D., 2003. Discriminative Training for Large Vocabulary Speech Recognition. Ph.D thesis, Cambridge University Engineering Department, 2003, pp. 1–172.

Reynolds, D., Quatieri, T., Dunn, R., 2000. Speaker verification using adapted Gaussian Mixture Models. Digital Signal Process. 10 (1–3), 19–41.

Schuller, B., Arsic, D., Wallhoff, F., Rigoll, G., 2006. Emotion recognition in the noise applying large acoustic feature sets. Speech Prosody, Dresden.

Schuller, B., Batliner, A., Seppi, D., Steidl, S., Vogt, T., Wagner, J., Devillers, L., Vidrascu, L., Amir, N., Kessous, L., Aharonson, V., 2007. The relevance of feature type for the automatic classification of emotional user states: Low level descriptors and functionals. INTERSPEECH 2007, 1–4, June.

Schuller, B., Steidl, S., Batliner, A., Feb 2009. The INTERSPEECH 2009 Emotion Challenge. In: Proceedings of Interspeech, Brighton, pp. 1–4.

Schwarz, P., Matejka, P., Cernocky, J., 2006. Hierarchical structures of neural networks for phoneme recognition. In: Proceedings of ICASSP 2006, Toulouse, pp. 325–328.

Seppi, D., Batliner, A., Schuller, B., Steidl, S., Vogt, T., Wagner, J., Devillers, L., Vidrascu, L., Amir, N., Aharonson, V., 2008. Patterns, prototypes, performance: classifying emotional user states. In: Proceedings of Interspeech.

Shriberg, E., Kajarekar, S., Scheffer, N., 2009. Does session variability compensation in speaker recognition model intrinsic variation under mismatched conditions? Interspeech Brighton.

Steidl, S., 2009. Automatic classification of emotion-related user states in spontaneous children's speech. Studien zur Mustererkennung, Bd. 28, ISBN 978-3-8325-2145-5, 1–260 (January).

Torres-Carrasquillo, P., Singer, E., Kohler, M., Greene, R., Reynolds, D., Jr, J.D., 2002. Approaches to language identification using Gaussian Mixture Models and shifted delta cepstral features. In: Seventh International Conference on Spoken Language Processing.

Ververidis, D., Kotropoulos, C., 2003. A state of the art review on emotional speech databases. In: Proceedings of 1st Richmedia Conference, pp. 109–119.

Vlasenko, B., Schuller, B., Wendemuth, A., Rigoll, G., 2007. Combining frame and turn-level information for robust recognition of emotions within speech. In: Proceedings of Interspeech, pp. 2249–2252.

Young, S., Evermann, G., Gales, M., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P., 2006. The htk book version 3.4.

# CONTOUR MODELING OF PROSODIC AND ACOUSTIC FEATURES FOR SPEAKER RECOGNITION

*Marcel Kockmann*[1 2]*, Lukáš Burget*[1]

[1]Speech@FIT, Brno University of Technology, Czech Republic
[2]Siemens AG, Corporate Technology, Munich, Germany
{kockmann|burget}@fit.vutbr.cz

## ABSTRACT

In this paper we use acoustic and prosodic features jointly in a long-temporal lexical context for automatic speaker recognition from speech. The contours of pitch, energy and cepstral coefficients are continuously modeled over the time span of a syllable to capture the speaking style on phonetic level. As these features are affected by session variability, established channel compensation techniques are examined. Results for the combination of different features on a syllable-level as well as for channel compensation are presented for the NIST SRE 2006 speaker identification task. To show the complementary character of the features, the proposed system is fused with an acoustic short-time system, leading to a relative improvement of 10.4%.

***Index Terms***— Speaker recognition, Prosody, GMM, Channel Compensation

## 1. INTRODUCTION

State-of-the-art systems for text independent speaker identification usually make use of acoustic short-time features in a Gaussian Mixture Model (GMM) framework with Universal Background Model (UBM) [1]. As these systems are strongly affected by session variability, new techniques have been successfully developed in the last few years to compensate for these channel effects [2]. Still, most acoustic systems do not make use of information from a higher level of speech, like the phonetic, prosodic or lexical layer. Different studies have shown that adding phonotactic- or prosodic characteristics to an acoustic baseline system can yield to a better overall performance, especially when a large amount of data is available per speaker [3]. Dehak *et al.* [4] also reported gain in recognition performance on shorter tasks, where only a few hundred feature vectors are available to train and test each speaker.

The work in this paper is based on the use of classical prosodic features like duration, pitch and energy in a syllable-like temporal context. The trajectories of each feature is continuously modeled over the time span of a syllable and is represented by coefficients from a discrete cosine transformation (DCT). Additionally we also capture the contour of acoustic features in form of Mel-frequency cepstral coefficients (MFCC) and form a single feature vector out of duration and pitch, energy and the MFCC contours. All these features are jointly modeled using a GMM. As this mixed feature vector will also be affected by variations in the channel, established techniques for the compensation of session variability are applied. Since each feature vector represents one syllable in the utterance, there are only a few hundred features per recording, which makes it hard to reliably estimate the channel factors that determine how far a model is shifted in the channel subspace. We will investigate if channel compensation in the model or in the feature domain is more appropriate for this small amount of feature vectors.

The performance of the proposed system is presented in terms of equal error rate for the text-independent NIST SRE 2006 speaker identification task [5].

The organization of the paper is as follows: section 2 describes the extraction of the syllable based features, including the basic features itself, the way the utterance is segmented into syllable-like units and based on this, the actual modeling of the temporal trajectory of the basic features. Section 3 briefly describes the algorithms used to perform the channel compensation. Section 4 presents the experiments and results obtained with the system and conclusions are given in section 5.

## 2. SYLLABLE BASED FEATURE CONTOURS

This section describes how a feature vector for each syllable is obtained by continuously modeling the temporal trajectory of various frame based features.

### 2.1. Basic features

Different basic features are extracted at 10-ms intervals. Pitch frequencies are computed with the Average Magnitude Difference Function from the Snack Sound Toolkit [6]. Snack is also used to obtain windowed log power values. All these features are extracted with Snacks default settings. Furthermore 12 Mel-frequency cepstral coefficients (20ms Hamming window, 23 bands in Mel filter bank) are generated.

### 2.2. Syllable segmentation

The segmentation into syllable-like units is based on the phonetically alignment from a phoneme recognizer with long temporal context [7]. We use a Hungarian recognizer, whose tokens are mapped to classes silence, consonant and vowel. Then each speech segment between two pauses is equally divided based on the number of vowels in this segment. Figure 1 shows how each vowel is considered as the nucleus of a syllable. In a second step, the estimated syllable boundary between two vowels can be shifted with regard to the measured pitch at the potential boundary candidates. This is done in order to preserve consecutive pitch contours that proceed for example from a vowel to a voiced consonant.

### 2.3. Contour modeling

*2.3.1. Pre-processing*

All basic features are pre-processed before actually modeling the temporal contour of them. Feature warping [8] (blind warping into normal distribution) is applied to all MFCCs and the logarithm is computed for the pitch frequencies. Finally, mean subtraction is applied to all features. Note that the mean was computed over the voiced parts of the whole utterance only (obtained by valid pitch). Small gaps (1 frame) in the pitch contour are smoothed by a median filter.

*2.3.2. Temporal trajectory*

The temporal contour of each feature can be approximated by a curve fitting tool, as shown in Figure 1. We use the first $n$ DCT bases to model the trajectory, which correspond to characteristics of the curve, like mean, slope and finer details. The contour is represented by its DCT coefficients in the feature vector. The advantage of using discrete cosine transformation instead of a simple polynomial curve fitting is, that mapping the contour segment to a fixed length is not necessary and that the coefficients are already decorrelated. As pitch may be undefined over parts of the syllable, one can consider different approaches to model the other features which are always defined within the syllable. In this work, jointly modeling the unvoiced and voiced part and modeling only the voiced part of each syllable is investigated for the other features.

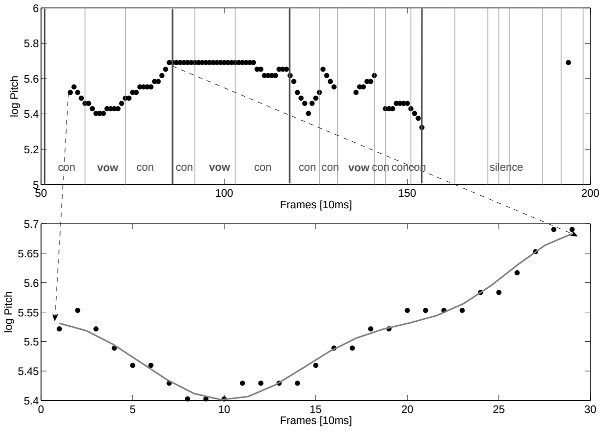


**Fig. 1**. *Example for pitch contour over syllable with three phonemes. Top: Original pitch values with phoneme and pseudo-syllable boundaries (horizontal lines). Bottom: Original (dotted line) and DCT approximated curve (solid line).*

### 2.4. Final feature vector

The number of voiced/unvoiced frames inside the syllable also serves as a discrete duration feature. The final feature vector for each syllable consists of the duration followed by the representation of the temporal contour for each basic feature like pitch, energy and MFCCs. Syllable segments that contain less frames than the number of DCT coefficients used to model the contour are omitted.

## 3. CHANNEL COMPENSATION

Prosodic features like pitch and energy shall be used along with acoustic features like MFCCs. Channel compensation has proved to be beneficial for both of these feature types [4]. Challenging is the use of channel compensation with relatively sparse feature vectors as it is the case here. For this purpose, eigenchannel compensation was performed in both, feature and model domain as it was proposed in [9] and [10]. This section gives a brief overview how the jointly used eigenchannel subspace was estimated as well as to the principles of the two different compensation techniques.

### 3.1. Eigenchannel Subspace

The eigenchannel subspace is a low dimensional representation of how the means of a GMM representing a speaker can be affected by changing channel. This subspace is estimated as described in [9]. Briefly, a corpus with multiple recordings for each speaker under various conditions is needed. After adapting the UBM to each training utterance, mean supervectors are formed by concatenating all mean vectors and dividing them by corresponding standard deviation. The eigenchannels are the eigenvectors of the average within-speaker covariance matrix. It is sufficient to keep only the the directions that cover most of the variability caused by channel effects (largest eigenvalues).

### 3.2. Eigenchannel Compensation in model and feature domain

Eigenchannel compensation in model domain is only applied to test conversations. During a single MAP-iteration, channel factors are estimated for the UBM as well as for each speaker model in test. These factors determine, how far each model is shifted towards the test-utterance in the directions defining the eigenchannel subspace. A simplified implementation for estimating the channel factors is used for computational efficiency as described in [9].

A more simplified approach of channel compensation leads to the possibility of shifting the features itself, rather than the models as proposed in [10]. One can assume to globally estimate the channel factors according only to the UBM. The change in means of the mixture component with the highest occupation probability is then applied to the feature vector itself. The channel compensated features can be used to train and test a standard GMM system.

## 4. EXPERIMENTS

### 4.1. Data

Experiments were performed on the core condition of the NIST 2006 speaker recognition evaluation (SRE) [5], which contains English trials only. The 1-side training 1-side test condition is considered, where approximately $2.5min$ of speech is available from a $5min$ telephone conversation to train each speaker and for each test trial. This set originally contains 462 female and 354 male training utterances (where multiple utterances can arise from one speaker) and 51448 test trials. Results are presented in terms of equal error rate (EER)[1]. The UBM model is trained on 7880 $5min$ utterances from the NIST 2004 and 2005 SRE data sets. The eigenchannel subspaces were estimated on 3399 sessions from 310 speakers (at least 8 sessions per speaker) from the NIST 2004 SRE training set. The same

[1] Note that evaluation key version 9 from NIST was used to measure the system performance.

corpus was used to normalize verification scores via z-norm [11] using 248 utterances.

### 4.2. Framework

The GMM framework used for the whole system is the same as used for an acoustic baseline system [9]. The gender-independent UBM is obtained by Expectation-Maximization (EM) Training and the speaker models are derived by MAP-Adaptation with $\tau = 19$. Discrete as well as continuous features are used within one feature vector, so variance flooring is crucial while EM training. Variances are floored to $1/100$ of the global variance. If not mentioned otherwise, all results are obtained with 256 Gaussians, no eigenchannel compensation and no z-norm.

### 4.3. Prosodic contour features

First experiments were performed with a classical prosodic feature vector, which comprises the duration of the syllable as well as the approximated pitch and energy contours, which are modeled with 6 DCT coefficients (minimal segment length is $60ms$). Results for different assortments of the feature vector are presented in Table 1. As can be seen it is most beneficial to use duration, pitch and energy jointly which also conforms to similar results in [4].

**Table 1**. *Different prosodic feature vectors with 6 coefficients per contour.*

| 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 |

As the feature vector will grow through the augmentation of MFCC features, we want to use the smallest number of coefficients to properly approximate the temporal contour in terms of recognition performance. Table 2 shows that modeling even finer details is not beneficial and that only a slight degradation has to be accepted by reducing the resolution to 4 DCT coefficients.

**Table 2**. *Pitch & Energy contours modeled by different number of DCT coefficients.*

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

The best performing 13-dimensional feature vector was also used to study the treatment of unvoiced parts within a syllable. Either the duration and the energy contour may correspond to the whole syllable or only to the voiced part. As can be seen in Table 3, it is beneficial to use only the voiced part of the syllable. Note also that the mean subtraction of the basic features in the pre-processing step is based only on the voiced parts as well. Using all speech segments as determined by the phoneme recognizer to compute the mean yields to much worse results.

**Table 3**. *Modeling whole syllable or only voiced part.*

| Feature Vector | EER [%] |
|---|---|
| whole Duration, Pitch & whole Energy Contour | 25.73 |
| voiced Duration, Pitch & voiced Energy Contour | 24.4 |

### 4.4. Expansion of feature vectors

For the following experiments, the number of DCT coefficients was reduced to 4. As the minimal segment length also is reduced to $40ms$, about 10% more feature vectors could be extracted for each utterance. This and additional feature warping of the energy coefficients reduced the EER to 22.3%, which serves as a reference for expanding the feature vector with MFCC contours.

In order to add a simple acoustic information, the prosodic feature vector was augmented with the means of 12 MFCCs over the syllable. This results in a drastic gain in recognition performance to 14.07%. The benefit of adding all coefficients for the MFCC contours can be seen in Table 4. Adding information about the temporal contour of all MFCCs yields to an EER of 9.87%, which is a relative improvement of 55% compared to the purely prosodic system. Even the contours of the higher MFCCs are beneficial and omitting them always results in worse performance (see also Table 4). Also the addition of the cepstral contours does not make the prosodic information negligible, as performance degrades to 10.63% for cepstral contours only.

**Table 4**. *Augmentation of prosodic feature vector (baseline: duration, pitch & energy contour). Contours are modeled with 4 coefficients, voiced parts only.*

| Feature Vector | Dim | EER [%] |
|---|---|---|
| Baseline | 9 | 22.3 |
| Baseline + 12 MFCC means | 21 | 14.07 |
| Baseline + 12 MFCC Contours | 57 | **9.87** |
| Baseline + 11 MFCC Contours | 53 | 10.14 |
| Baseline + 10 MFCC Contours | 49 | 10.57 |
| Baseline + 9 MFCC Contours | 45 | 11.22 |
| Baseline + 8 MFCC Contours | 41 | 11.27 |
| 12 MFCC Contours | 48 | 10.63 |

### 4.5. Channel Compensation

The effectiveness of eigenchannel compensation in model and feature domain was investigated for a system trained on a 57-dimensional vector containing duration and the temporal trajectories for pitch, energy and 12 cepstral coefficients. 10 eigenchannels were used in the experiments. Note that only approximately 500 feature vectors are available in this syllable-framework to estimate the channel factors that determine the compensation of each utterance. Table 5 shows the effect of the channel compensation for GMMs with different number of Gaussians. For small models with only 32 Gaussians, the channel factors can be estimated quite well and the compensation in model as well as in feature domain results in 30% relative improvement, while for a model with 512 Gaussians, the gain is only about 5%. Unfortunately the small models perform much worse before applying the channel compensation, and EER is still worse after eigenchannel adaptation. However, for the model with 256 Gaussians the EER could still be reduced by $11\%$ to $8.74\%$, even with this small amount of data.

**Table 5**. *Effects of channel compensation for different sized GMMs (10 Eigenchannels) in EER [%].*

| # of Gaussians | No CC | Model Domain | Feature Domain |
|---|---|---|---|
| 512 | 9.44 | 9.06 | 9.06 |
| 256 | 9.87 | 8.8 | 8.74 |
| 128 | 10.89 | 8.8 | 8.75 |
| 64 | 12.35 | 9.3 | 9.3 |
| 32 | 14.88 | 10.41 | 10.42 |

Eigenchannel compensation in feature domain bears the opportunity to compensate the features on an eigenchannel subspace created on a smaller UBM and do the model training and evaluation with a larger GMM. This technique assumes that the properly estimated channel directions and channel factors also fit for the bigger GMM. In our experiments the features were compensated on GMM sizes where the standard compensation showed adequate performance. These compensated features were used to train model sizes that performed best without channel compensation. As can be seen in Table 6, this approach to handle the sparse data results in better performance than the normal eigenchannel adaptation. The relative improvement compared to the standard compensation is $6\%$ and $8\%$ for the GMM sizes 256 and 512, respectively.

**Table 6**. *Different sized models with features compensated on smaller Eigensubspace (sizes in # of Gaussians).*

| Speaker UBM | Subspace UBM | EER [%] |
|---|---|---|
| 512 | 128 | 8.31 |
| 512 | 64 | 8.36 |
| 256 | 128 | 8.2 |
| 256 | 64 | 8.36 |
| 128 | 64 | 8.9 |

### 4.6. Combination with acoustic baseline system

Finally the complementary information of this syllable-based system to a short-time acoustic system is to be investigated by fusing it with a state-of-the-art acoustic GMM system (2048 Gaussians, 13 MFCCs, feature warping, single-, double- and triple deltas, HLDA and eigenchannel adaptation) [9]. Before the fusion, z-norm is applied to the proposed syllable-based system with channel compensation in feature domain, which yields to an EER of $7.66\%$. Table 7 shows that combining it with the baseline system by a linear fusion [12] yields to a relative improvement of $10.4\%$ to an overall EER of $2.75\%$.

**Table 7**. *Fusion of best performing syllable-based system with acoustic baseline.*

| System | EER [%] |
|---|---|
| Duration, Pitch, Energy & 12 MFCC Contours<br>256 Gaussians, z-norm | 7.66 |
| 2048 Gaussians acoustic baseline | 3.07 |
| Fusion | 2.75 |

## 5. CONCLUSIONS

We have shown that syllable based prosodic feature vectors can be successfully expanded and jointly modeled with acoustic cepstral features by the use of DCT coefficients to represent the temporal contour of each phonetically motivated segment. The addition of cepstral contours achieves over $50\%$ improvement compared to a classical prosodic system with duration, pitch and energy only. Without any compensation for session variability, the performance of such a system is comparable to a frame-based acoustic system and comprises complementary information through different kinds of features like pitch and a different temporal context. As the effect of channel compensation (frame-based acoustic systems improve relatively about $50\%$) decreases for the proposed system due to the small amount of features in the test utterance, an approach could be presented to gain more improvement through the use of channel compensation in feature domain, where features are compensated through a smaller and more robust eigenchannel subspace. When combining this system with best-performing baseline acoustic system it results in a $10.4\%$ improvement of overall performance.

## 6. ACKNOWLEDGMENTS

This work was partly supported by European projects AMIDA (FP6-033812), Caretaker (FP6-027231) and MOBIO (FP7-214324), by Grant Agency of Czech Republic under project No. 102/08/0707 and by Czech Ministry of Education under project No. MSM0021630528. The hardware used in this work was partially provided by CESNET under project No. 201/2006. Lukas Burget was supported by Grant Agency of Czech Republic under project No. GP102/06/383.

## 7. REFERENCES

[1] Reynolds, D. A. et al., “Speaker Verification Using Adapted Gaussian Mixture Models”, Digital Signal Processing 10, 19-41 (2000).

[2] Kenny, P. and Dumouchel, P., “Disentangling speaker and channel effects in speaker verification”, in Proc. ICASSP, 2004, pp. 37–40.

[3] Reynolds, D. A. et al., “The SuperSID Project: Exploiting High-level Information for High-accuracy”, Acoustics, Speech, and Signal Processing, 2003. Proceedings.

[4] Dehak, N. et al., “Modeling Prosodic Features With Joint Factor Analysis for Speaker Verification”, in Audio, Speech, and Language Processing, September 2007, Volume 15. pp. 2095–2103.

[5] “The NIST Year 2006 Speaker Recognition Evaluation Plan”, Online on: http://www.nist.gov/speech/tests/spk/2006.

[6] Sjölander, K., “The Snack Sound Toolkit”, Online on: http://www.speech.kth.se/snack.

[7] Schwarz, P. et al., “Hierarchical structures of neural networks for phoneme recognition”, in Proceedings of ICASSP, Toulouse, 2006.

[8] Pelecanos, J. and Sridharan, S., “Feature Warping for Robust Speaker Verification”, in proc of A Speaker Odyssey, 2001.

[9] Burget, L. et al.,“Analysis of feature extraction and channel compensation in GMM speaker recognition system,” in IEEE Trans. on Audio, Speech and Language Processing, September 2007.

[10] Castaldo, F., et al.,“Compensation of Nuisance Factors for Speaker and Language Recognition”, in IEEE Trans. on Audio, Speech and Language Processing, September 2007, Volume 15. pp. 1969–1978.

[11] Auckenthaler, R. et al., “Score normalization for text-independent speaker verification systems”, in Digital Signal Processing, 10/2000.

[12] Brümmer, N. and Preez, J. d., “Application-Independent Evaluation of Speaker Detection”, Computer Speech and Language, 2005, Online on: http://www.dsp.sun.ac.za/ nbrummer/focal.

# Chapter 5

# Applications of i-vectors

The publications included in this section demonstrate the applicability of i-vectors to other than speaker recognition problem (sections 5.1, 5.2, 5.3, 5.7). Conceptually new approaches to fusion (section 5.5) and to dicriminative training 5.4 of speaker verification systems are described, all building on the concept of i-vectors. Originally, i-vectors were proposed to represent sequences of continuous feature vectors. The publications extending this concept to sequences of discrete features are also included. For this purpose, a new subspace multinomial model (section 5.6) and subspace n-Gram model (section 5.7) were proposed. A publication that introduces a nowadays popular technique for i-vector based discriminative adaptation of speech recognition system is also included (section 5.3).

# Language Recognition in iVectors Space

*David Martínez*[1], *Oldřich Plchot*[2], *Lukáš Burget*[2], *Ondřej Glembek*[2] *and Pavel Matějka*[2]

[1]Aragon Institute for Engineering Research (I3A), University of Zaragoza, Spain
[2]Speech@FIT, Brno University of Technology, Czech Republic

david@unizar.es, {iplchot,burget,glembek,matejkap}@fit.vutbr.cz

## Abstract

The concept of so called iVectors, where each utterance is represented by fixed-length low-dimensional feature vector, has recently become very successfully in speaker verification. In this work, we apply the same idea in the context of Language Recognition (LR). To recognize language in the iVector space, we experiment with three different linear classifiers: one based on a generative model, where classes are modeled by Gaussian distributions with shared covariance matrix, and two discriminative classifiers, namely linear Support Vector Machine and Logistic Regression. The tests were performed on the NIST LRE 2009 dataset and the results were compared with state-of-the-art LR based on Joint Factor Analysis (JFA). While the iVector system offers better performance, it also seems to be complementary to JFA, as their fusion shows another improvement.

**Index Terms**: Acoustic Language Recognition, iVectors, Joint Factor Analysis.

## 1. Introduction

Joint Factor Analysis (JFA) [15], which is a statistical model originally proposed for Speaker Recognition, has become very successful also for acoustic Language Recognition (LR) [3, 2]. The idea behind JFA is to consider not only the inter-class variability in the space of model parameters (we have different model parameters for different languages in LR), but also the inter-session variability (parameters for a language can change from utterance to utterance because of the differences in channel, speaker, etc.). We will refer to the latter variability simply as *channel variability*. When the likelihood of a test utterance is evaluated for a certain language, the corresponding model is adapted to the channel of that test utterance. This is done by finding the point MAP (or ML) estimate of a low-dimensional latent variable vector - *channel factors*, which are coordinates in a highly channel-variable subspace of the model parameter space.

Recently, systems based on iVectors [4, 16] have provided superior performance in speaker recognition. iVector is a fixed-length low-dimensional vector, which is extracted for each utterance based on the JFA-like idea of estimating latent variables corresponding to high variability subspace. The principal difference from JFA is that we are not interested in evaluating the adapted model. Instead, the latent variables - iVectors - are used as features for another (possibly very simple) classifier. Also, the underlying model for iVector extraction does not attempt to separate inter-class and channel variability. Instead, it considerers only single *total variability* subspace corresponding to both sources of variability. The advantage is that the model for iVector extraction can be trained in unsupervised manner (without providing speaker or language identities for speaker or language recognition respectively). On the other hand, iVector contains information about both the class and the channel; this has to be taken into account in the following classifier.

Inspired by the success of iVectors in speaker recognition, we apply the same idea in the context of language recognition in this work. As a classifier in the iVector space, we use the linear generative model, where the distribution of iVectors for each language is Gaussian with full covariance matrix shared across languages. This model is analogue to Probabilistic Linear Discriminant Analysis (PLDA) [1], which is currently the most successful model for modeling iVectors in speaker recognition [16, 13]. Unlike in PLDA, we do not need to explicitly model distributions of class means. We deal here only with a closed-set problem, where means for a limited number of classes (languages) can be robustly obtained as the ML estimates. However, note that the PLDA approach, thanks to that inter-class distribution modeling, could be useful when dealing with an open-set LR problem, where also unknown out-of-set languages have to be detected.

Low dimensionality of iVectors makes it also convenient to apply discriminative classifiers. We have experimented with linear Support Vector Machines (SVM) and Logistic Regression in combination with Nuisance Attribute Projection (NAP) [11] as a channel compensation technique.

The performance of the proposed techniques is compared with state-of-the-art JFA based system on the NIST LRE 2009. On 30s condition, the best performing individual system is iVector based generative model, where $C_{avg} = 0.0188$ corresponds to 7% improvement over the JFA baseline. Further improvements (up to 18% over the JFA baseline) can be obtained by fusing the JFA and iVector based systems.

Note that in [9], another iVector based approach is applied to phonotactic language recognition, where recently proposed Subspace Multinomial Model [5] is used to extract iVector from phone n-gram counts.

The rest of the paper is organized as follows: in Section 2, iVectors fundamentals are revisited; in Section 3, the classifiers used for the experimentation are reviewed; in Section 4, the experimental setup is described; in Section 5, the results are presented; and in Section 6, the conclusions are derived.

## 2. iVectors

The iVector approach has become state-of-the-art in the speaker verification field [4] and, in this work, we show that it can be successfully applied also to language recognition. The approach provides an elegant way of reducing high-dimensional sequential input data to a low-dimensional fixed-length feature vector while retaining most of the relevant information. The main idea is that the language- and channel-dependent supervectors of concatenated Gaussian Mixture Model (GMM) means can be modeled as

$$\mathbf{M} = \mathbf{m} + \mathbf{T}\mathbf{w}, \tag{1}$$

where $\mathbf{m}$ is the language- and channel-independent component of the mean supervector, $\mathbf{T}$ is a matrix of bases spanning the subspace covering the important variability (both speaker- and session-specific) in the supervector space, and $\mathbf{w}$ is a standard-normally distributed latent variable. For each observation sequence representing an utterance, our iVector is the Maximum A Posteriori (MAP) point estimate of the latent variable $\mathbf{w}$. Our iVector extractor training procedure is based on the efficient implementation suggested in [7].

# 3. Classifiers

### 3.1. Generative model

In the case of the generative model, distribution of iVectors for each language is modeled by a Gaussian distribution, where full covariance matrix is shared across all languages. For an iVector $\mathbf{w}$ corresponding to a test utterance, we evaluate log-likelihood for each language as:

$$\ln p(\mathbf{w}|l) = -\frac{1}{2}\mathbf{w}^T\boldsymbol{\Sigma}^{-1}\mathbf{w}+\mathbf{w}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_l-\frac{1}{2}\boldsymbol{\mu}_l^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_l+const,$$

where $\boldsymbol{\mu}_l$ is the mean vector for language $l$, $\boldsymbol{\Sigma}$ is the common covariance matrix and $const$ is a language- and iVector-independent constant. If the log-likelihoods $\ln p(\mathbf{w}|l)$ were directly used to decide about the language (or estimate the posterior probability of a language), the quadratic term $\mathbf{w}^T\boldsymbol{\Sigma}^{-1}\mathbf{w}$ could be ignored as it is independent of the class thanks to the shared covariance matrix. This would lead to linear classifier as the remaining terms are only linear in $\mathbf{w}$. In our case, however, the log likelihoods are used as inputs to another classifier, the calibration back-end described in section 4.3. For this reason, we include the quadratic term, and thus, we avoid the iVector (utterance) dependent shift in our scores.

### 3.2. Discriminative Classifiers

We have also experimented with discriminative linear classifiers: linear Support Vector Machines (SVM) and Logistic Regression with L2 regularization. In both cases, binary classifiers are trained and one-versus-all strategy is used to obtain scores for all languages. We use implementations from LIBSVM [10] and LIBLINEAR [12] for SVM and logistic regression, respectively. Although, we have used binary logistic regression in our experiments, our problem could be addressed more directly using a single multi-class logistic regression classifier. For example, the experiments in [3], where multi-class logistic regression was applied to recognize languages from GMM mean supervectors, can be now carried out in iVector space with significantly reduced computational cost and space complexity.

# 4. Experimental Setup

### 4.1. Training and Development Data

Our training data were taken from the same databases as in [2]: Callfriend, Fisher English Part 1 and 2, Fisher Levantine Arabic, HKUST Mandarin, Mixer (data from NIST SRE 2004, 2005, 2006, 2008). We have defined two sets with data from the 23 NIST LRE 2009 target languages only: the first contains all the utterances in the databases for these languages and it is further denoted *full*. The second contains a maximum of 500 utterances per language (we do not have 500 utterances for all languages), and it is further denoted *balanced*. For training the iVector extractor, the full dataset has been taken, but no degradation in performance was seen when using the balanced one. For training the classifiers, the balanced dataset has been taken, because it was found that having equal amount of data per class leads to lower error rates.

The calibration back-end described in section 4.3 was trained on development dataset, which comprises data from NIST LRE 2007, OGI-multilingual, OGI 22 languages, Foreign Accented English, SpeechDat-East, Switch Board and Voice of America radio broadcast. Only data of the 23 target languages are used. This set was based on segments of previous NIST LRE evaluations plus additional segments extracted from CTS, VOA3 and human-audited VOA2 data, not contained in the training dataset, and is the same as in [2].

### 4.2. Feature Extraction

Standard 7 Mel Frequency Cepstral Coefficients (MFCC) (including $C_0$) are used. Vocal Tract Length Normalization (VTLN) [8] and Cepstral Mean and Variance Normalization is applied in MFCC computation. Then, Shifted Delta Cepstral (SDC) coefficients [6] with usual 7-1-3-7 configuration are obtained, and concatenated to MFCCs, to obtain a final feature vector of 56 coefficients. For each utterance, the corresponding feature sequence is finally converted to an iVector using an iVector extractor based on a GMM with 2048-components trained on pooled features from all 54 languages included in our training data.

### 4.3. Calibration Back-end

For calibration and fusion, a Gaussian Back-end followed by a Discriminative Multi-Class Logistic Regression is used to post-process scores obtained from the described classifiers. Note that the Gaussian Back-end is essentially the same model as our generative classifier. However, its inputs are the scores from the classifiers described above rather than the iVectors. Also, it is trained on the separate development dataset to obtain well-calibrated scores.

# 5. Results

All results are for the closed-set condition. We use the NIST LRE 2009 dataset, which contains 23 target languages, and files of 3, 10 and 30 s. Results are shown in terms of $C_{avg} \times 100$ defined in the NIST LRE 2009 Evaluation Plan[1]. Since at the output of the backend well-calibrated log-likelihoods are obtained, the threshold is set analytically.

### 5.1. Results for Generative Linear Classifier

In Table 1, we show the effect of iVectors dimensionality for three conditions corresponding to the three nominal durations of test utterances (3, 10 and 30 s). We can see that the appropriate iVector dimensionality is 600. A lower dimensionality does not give the same level of accuracy and higher dimensionality does not offer further improvements, while the computational complexity is increased. Also, duration-independent (DI) calibration back-end is compared to the duration-dependent (DD) back-end, where a separate back-end is trained for each condition. As we can see, no significant difference between DI and DD back-end for the 30 s condition is found. However, for the 3 and 10 s conditions, the DI back-end performs better. This

[1] http://www.itl.nist.gov/iad/mig/tests/lre/2009/LRE09_EvalPlan_v6.pdf

| Condition | 200D | 300D | 400D | 500D | 600D | 700D |
|---|---|---|---|---|---|---|
| 3 s DI | 14.78 | 14.54 | 14.35 | 14.30 | **14.10** | 14.12 |
| 3 s DD | 16.29 | 15.87 | 15.63 | 15.50 | 15.29 | 15.25 |
| 10 s DI | 4.63 | 4.33 | 4.26 | 4.14 | **4.04** | 4.05 |
| 10 s DD | 5.55 | 5.25 | 5.11 | 4.90 | 4.76 | 4.79 |
| 30 s DI | 2.29 | 2.07 | 1.94 | 1.94 | 1.91 | 2.01 |
| 30 s DD | 2.36 | 2.08 | 1.88 | 1.90 | **1.88** | 1.93 |

Table 1: $C_{avg} \times 100$ *for the generative model with 200 to 700 dimensions, for the 3, 10 and 30 s conditions, and for the DI and DD back-ends*

indicates that scores obtained from the generative model are independent of the duration of the test utterances and we can benefit from training the back-end on larger amount of data pooled from the three conditions. For this reason, only the DI back-end is used in the remaining experiments.

In speaker recognition, significantly improved performance was observed when the dimensionality of iVectors was reduced by LDA and/or length of each iVector was normalized to unity [14] prior to applying the PLDA model. In Table 2, we can see that none of these techniques leads to an improvement in LR. The maximum number of useful dimensions that LDA can identify is the number of classes minus one. Since we have only 23 target languages, iVectors are reduced to 22 dimensions when applying LDA. Note that, since LDA and the generative model are both based on the same assumption of the common within-class covariance matrix, LDA dimensionality reduction would not have any effect if the classification decision was based directly on the generative model (for similar reasons as described in section 3.1). However, LDA causes utterance-dependent shifts to the likelihood scores (common to all classes) corresponding to the discarded dimensions, which makes the difference when using the generative model in conjunction with the following back-end.

| Condition | Generative | +NORM | +LDA |
|---|---|---|---|
| 3 s | **14.10** | 14.57 | 14.41 |
| 10 s | **4.04** | 4.32 | 4.13 |
| 30 s | **1.91** | 2.03 | 1.96 |

Table 2: $C_{avg} \times 100$ *for the iVectors and generative models*

### 5.2. Results for Discriminative Classifiers

First, we carried out experiments to find appropriate regularization constant for both SVM and logistic regression. Figure 1 and Figure 2 show performance obtained with SVM and logistic regression for different values of regularization parameter $C$ as defined in LIBSVM and LIBLINEAR (smaller $C$ leads to more aggressive regularization). The optimal performance was obtained with 400 dimensional iVectors and C=0.001 in the case of SVM, and with 600 dimensional iVectors and C=0.01 in the case of logistic regression. The following results are reported for these configurations.

In Tables 3 and 4, results obtained with SVM and logistic regression are shown. For both classifiers, we also experimented with three modifications. The first one is the application of Nuisance Attribute Projection [11], which projects *N* directions with the largest channel variability out of the iVectors. The second modification is the LDA dimensionality reduction of iVectors applied in the same way as in the case of the generative classifier. The third modification is iVector length normalization followed by LDA. As we can see, better results are generally obtained with logistic regression, where particularly good performance is obtained with NAP and with LDA (without iVector normalization).

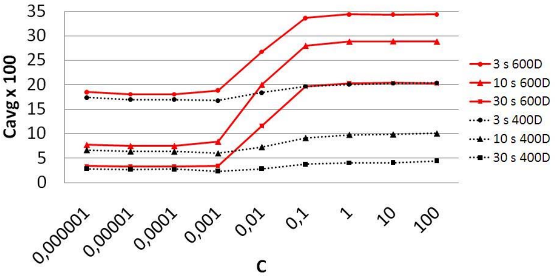


Figure 1: *Tuning of C value for SVM with iVectors of dimension 400 and 600 with the DI back-end*

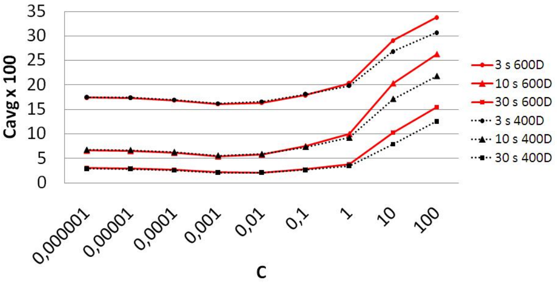


Figure 2: *Tuning of C value for logistic regression with iVectors of dimension 400 and 600 with the DI back-end*

Note that LDA dimensionality reduction and NAP are very similar techniques when applied in iVector space. First, NAP projects out the high channel variability directions while preserving the original dimensionality of iVectors. Although this is unnecessary with low dimensional iVectors, where appropriate linear transformation can be applied to remove the corresponding dimensions, just like in the case of LDA. Furthermore, the iVector extractor is trained in such a way that iVectors (at least those corresponding to training utterances) are standard normal distributed (i.e. variance of iVectors is one in all directions). Therefore, the directions with the largest ratio between across-class and within-class variance (preserved by LDA) are also the directions with the smallest within-class variance (preserved by NAP). However, unlike in the case of LDA, NAP allows us to preserve more than 22 dimensions, which might be found useful by the discriminative classifier. The search for optimal dimensionality of channel subspace in NAP is shown in Figure 3, for both SVM and logistic regression (only the 10 s condition is plotted for a clearer representation, the 3 s and 30 s condition follow the same trend). In both cases the optimal dimension is $N = 60$, and this is the dimension used to run experiments.

### 5.3. Comparison with JFA and fusion

Table 5 shows results for JFA (as described in [3]), for the best performing iVector based systems, and for fusion of both approaches. Both generative and discriminative classifiers based on iVectors outperform the state-of-the art JFA system and fusion of JFA and iVector based systems leads to additional improvements. It is interesting to see that most of the improve-

| Condition | SVM | +NAP | +LDA | +NORM+LDA |
|---|---|---|---|---|
| 3 s | 15.84 | 15.71 | 14.99 | **14.66** |
| 10 s | 5.16 | 5.00 | 4.56 | **4.39** |
| 30 s | 2.24 | **2.03** | 2.10 | 2.28 |

Table 3: $C_{avg} \times 100$ *obtained with SVM classifier. Experiments with 400 dimensional iVectors*

| Condition | LgR | +NAP | +LDA | +NORM+LDA |
|---|---|---|---|---|
| 3 s | 15.14 | **13.86** | 14.05 | 14.25 |
| 10 s | 4.88 | 4.06 | **4.03** | 4.17 |
| 30 s | 2.05 | **1.92** | 1.93 | 2.17 |

Table 4: $C_{avg} \times 100$ *obtained with logistic regression classifier. Experiments with 600 dimensional iVectors*

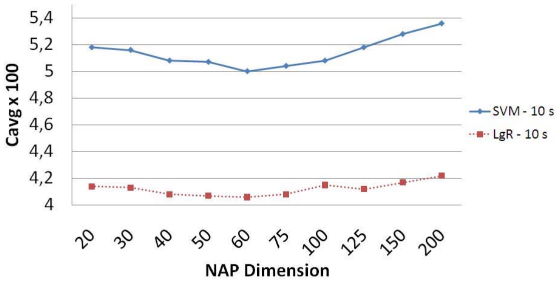


Figure 3: *Tuning of NAP dimensionality for SVM with 400D iVectors and LgR with 600D iVectors, for the 10 s condition*

| System | JFA | Generative | SVM+LDA | LgR+LDA | Fus1 | Fus2 |
|---|---|---|---|---|---|---|
| 3 s | 14.57 | 14.10 | 14.66 | 14.05 | 13.88 | **13.81** |
| 10 s | 4.89 | 4.04 | 4.39 | 4.03 | 3.86 | **3.82** |
| 30 s | 2.02 | 1.88 | 2.10 | 1.90 | 1.70 | **1.66** |

Table 5: $C_{avg} \times 100$ *for the JFA system from [3], the best performing iVector based systems, and for fusion of both approaches:*
*Fus1: fusion of JFA and Generative*
*Fus2: fusion of JFA, Generative, SVM+LDA and LgR+LDA*

ment is obtained when fusing JFA with only one single iVector system based on generative model and that fusion of all the individual systems in Table 5 leads only to insignificant additional $C_{avg}$ reductions.

## 6. Conclusions

We have introduced a novel approach for language recognition. Three classifiers (linear generative model, SVM and logistic regression) have been tested in the iVector space, and all outperform the state-of-the-art JFA system. Very simple and fast classifier based on linear generative model provides excellent performance over all conditions. The advantage of this classifier is also its scalability: addition of a new language only requires estimating the mean over the corresponding iVectors. Most of the computational load is in the iVector generation. Hence, as a next step, we will try to obtain iVectors from the utterances and the corresponding sufficient statistics in a more direct way.

## 7. Acknowledgements

This work was partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. D10PC20015, and by Czech Ministry of Education project No. MSM0021630528. It was done during an internship of David Martínez at Brno University of Technology funded by the Spanish Ministry of Science and Innovation under project TIN2008-06856-C05-04.

## 8. References

[1] Simon J. D. Prince and James H. Elder, “Probabilistic Linear Discriminant Analysis for Inference About Identity”, in Proc. 11th International Conference on Computer Vision, Rio de Janeiro, Brazil, Oct. 2007.

[2] Z. Jančik et al., “Data Selection and Calibration Issues in Automatic Language Recognition - Invesigation with BUT-AGNITIO NIST LRE 2009 System”, in Proc. Odyssey 2010 - The Speaker and Language Recognition Workshop, Brno, CZ.

[3] N. Brümmer, A. Strasheim, V. Hubeika, P. Matějka, P. Schwarz, J. Černocký, “Discriminative Acoustic Language Recognition via Channel-Compensated GMM Statistics”, in Proc. Interspeech 2009, Brighton, GB.

[4] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel and P. Ouellet, “Front-End Factor Analysis for Speaker Verification”, IEEE Trans. on Audio, Speech and Language Processing, vol. 19, pp. 788-798, May 2011.

[5] M. Kockmann et al., “Prosodic speaker verification using subspace multinomial models with intersession compensation”, in Proc. Interspeech, Tokyo, 2010

[6] P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, and J.R. Deller Jr., “Approaches to Language Identification using Gaussian Mixture Models and Shifted Delta Cepstral Features”, in Proc. International Conferences on Spoken Language Processing, Sept. 2002.

[7] O. Glembek, L. Burget, P. Matějka, M. Karafiat, P. Kenny, “Simplification and Optimization of i-vector Extraction”, accepted to ICASSP 2011, Prague.

[8] L. Weling, S. Kanthak and H. Ney, “Improved Methods for Vocal Tract Normalization”, in Proc. ICASSP 1999, Phoenix.

[9] M. Soufifar et al., “iVector Based Approach to Phonotactic Language Recognition”, submitted to Interspeech 2011.

[10] Chin-Chung Chang and Chih-Jen Lin, “LIBSVM: a Library for Support Vector Machines”, 2001, http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[11] W.M. Campbell, D.E. Sturim, D.A. Reynolds, A. Solomonoff, “SVM Based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation”, IEEE International Conference on Acoustics, Speech, and Signal Processing, Toulouse, 2006, vol. 1, pp. 97-100.

[12] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: A Library for Large Linear Classification”, Journal of Machine Learning Research 9 (2008), 1871-1874. http://www.csie.ntu.edu.tw/ cjlin/liblinear.

[13] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matějka, N. Brümmer, “Discriminatively Trained Probabilistic Linear Discriminant Analysis for Speaker Verification”, accepted to ICASSP 2011, Prague.

[14] D. García-Romero and C. Espy-Wilson, “Analysis of i-vector Length Normalization in Speaker Recognition Systems”, submitted to Interspeech 2011.

[15] P. Kenny et al., “Joint Factor Analysis versus Eigenchannes in Speaker Recognition”, IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, pp. 1435-1447, May 2007.

[16] P. Kenny, “Bayesian Speaker Verification with Heavy-Tailed Priors”, in proc. of Odyssey 2010 - The Speaker and Language Recognition Workshop, Brno, CZ.

# IVECTOR-BASED PROSODIC SYSTEM FOR LANGUAGE IDENTIFICATION

*David Martínez*[1], *Lukáš Burget*[2], *Luciana Ferrer*[2], *Nicolas Scheffer*[2]

[1]Aragon Institute for Engineering Research (I3A), University of Zaragoza, Spain
[2]Speech Technology and Research Laboratory, SRI International, Menlo Park, CA, USA

## ABSTRACT

Prosody is the part of speech where rhythm, stress, and intonation are reflected. In language identification tasks, these characteristics are assumed to be language dependent, and thus the language can be identified from them. In this paper, an automatic language recognition system that extracts prosody information from speech and makes decisions about the language with a generative classifier based on iVectors is built. The system is tested on the NIST LRE09 dataset. The results are still not comparable to state-of-the-art acoustic and phonotactic systems. However, they are promising and the fusion of the new approach with an iVector-based acoustic system is found to bring further improvements over the latter.

***Index Terms—*** Language Identification, Prosody, iVectors, Joint Factor Analysis.

## 1. INTRODUCTION

In recent years, we have seen great improvements in acoustic and phonotactic language identification (LID) systems. Among the most popular modeling techniques used in acoustic systems are joint factor analysis (JFA) [2] and iVectors [1], which are usually applied to model spectral features such as mel frequency cepstral coefficients (MFCC). In contrast, phoneme n-gram statistics are modeled in order to recognize languages in phonotactic approaches [3, 4].

Several approaches have been also investigated to extract prosodic information from speech and employ it in LID systems. In [6], the authors extract a set of features based on the three components of prosody: rhythm, stress, and intonation. However, the extraction procedure is computationally expensive since an automatic speech recognition (ASR) system is required. In [7], pitch contours are approximated using Legendre polynomials over long temporal intervals, which seems to be logical and useful for prosody modeling. This approach has also been recently adopted for speaker identification (SID) [8, 9, 10], where pitch contours and also energy contours are approximated using linear combination of Legendre polynomials over syllable or syllable-like units. The regression coefficients together with durations of corresponding segments are the features describing the three characteristics of prosody.

When modeling prosodic features for SID, different techniques have been proposed in the literature [8, 9, 10, 11, 12]. Until recently, one of the most popular approaches was to use a standard JFA model [8, 10]. Recently, the standard iVector approach [14], initially proposed to model MFCC features, was tested on polynomial coefficient prosodic features [11], showing remarkable performance on a speaker verification task, comparable to that obtained using the JFA approach. Note that these approaches are applicable only to features that are always defined and are relatively low-dimensional, like the polynomial coefficient features described above. For more complex sets of features, another subspace modeling technique called the subspace multinomial model (SMM) [12] was introduced, which models the vector of weights from a background Gaussian mixture model (GMM) that takes into account probabilities of undefined values. Recently, SMM-based iVectors were also successfully used as low-dimensional representations of n-gram counts in a phonotactic LID system [5].

In our work, we adopt the standard iVector paradigm [14] to model the prosodic polynomial features for LID, and create a classification system similar to the one from [1], where an iVector system is built based on acoustic features, and a generative Gaussian model for each of the languages with a shared covariance matrix is used as the classifier. Our systems are tested on the NIST LRE 2009 dataset [16], on which no previous results based on prosodic features are available. We hope that this can be useful as a baseline for future research on this topic.

The rest of the paper is organized as follows: in Section 2, the prosodic feature extraction process is described; in Section 3, the generative Gaussian LID system based on iVectors is revised; in Section 4, the experimental setup and results are shown; in Section 5, the conclusions are drawn.

## 2. PROSODIC FEATURE EXTRACTION

### 2.1. Pitch and Energy Contour Extraction

Our prosodic features carry information about the evolution of pitch and energy along time. To extract pitch and energy contours we use The Snack Sound Toolkit [15]. The pitch and energy values are converted to log domain, to simulate human perception. In the next step, energy is normalized by subtracting its maximum value in the log scale. This makes it more robust to language-independent phenomena such as channel variations. The log pitch values are normalized by subtracting mean and dividing by standard deviation estimated over each recording. In SID no normalization of pitch is required, since the absolute value contains information about the speaker. In LID, we are interested only in the information about the language and we believe that pitch normalization reduces the unwanted across-speaker variability. We have also experimented with only mean normalization, which resulted in very similar performance to mean and variance normalization, and for this reason, only results for mean and variance normalization will be shown.

### 2.2. Segment Definition

After extracting pitch and energy contours for whole speech recordings, every recording is divided into segments and coefficients describing pitch and energy contours are extracted for each such segment. In [10], different segment definitions were tested and segmentation based on syllables detected using an ASR system was found to perform the best. Since the language is unknown in the case of LID, we wanted to avoid the use of ASR. Therefore, we experimented with the other two segment definitions proposed in [10]: segment boundaries defined by energy valleys and fixed-length segments. For the energy valley based segments, segment boundaries are determined by local minima in the energy contour. This approach tries to find syllable boundaries in a very simple way. In the case of fixed-length segments, the signal is split into segments of 200 ms with an overlap of 150 ms. Compared to the segment length of 300 ms proposed in [10], our segments are closer to the average syllable duration of 120 ms. Also, shorter segments and larger overlap allow us to obtain more training examples for languages with small amounts of training data.

### 2.3. Contour Modeling

For each segment, we drop all unvoiced frames for which no pitch was detected. Then pitch and energy contours are approximated by linear combination of Legendre polynomials as

$$f(t) = \sum_{i=0}^{M} a_i P_i(t) \tag{1}$$

where $f(t)$ is the contour being modeled and $P_i(t)$ is the $i$ Legendre polynomial. Each coefficient $a_i$ represents a characteristic of the contour shape: $a_0$ corresponds to the mean, $a_1$ to the slope, $a_2$ to the curvature, and higher order represents more precise detail of the contour. In our implementation, Legendre polynomials of order 5 give six coefficients for pitch and six for energy.

Finally, 13-dimensional feature vectors are obtained by augmenting the coefficients with the number of voiced frames in the segment. Thus, we can consider that our features contain information of the three components of prosody: intonation in the pitch, rhythm in the duration, and stress in both the energy and in the duration. These are the features used to build our GMM universal background model (UBM). Supervectors of Baum-Welch statistics can then be estimated for each utterance, as in [14]. They are of dimension 13 times the number of Gaussians in the UBM.

## 3. IVECTORS AND CLASSIFICATION

### 3.1. iVector Extraction

The idea behind the iVector approach is that the language- and channel-dependent supervectors of concatenated GMM means can be modeled as

$$\mathbf{M} = \mathbf{m} + \mathbf{T}\mathbf{w}, \tag{2}$$

where $\mathbf{m}$ is a language- and channel-independent supervector of concatenated UBM means, $\mathbf{T}$ is a matrix of bases spanning the subspace covering the important variability (both language- and session-specific) in the supervector space, and $\mathbf{w}$ is a standard normally distributed latent variable. For each observation sequence representing an utterance, our iVector is the maximum a posteriori (MAP) point estimate of the latent variable $\mathbf{w}$. For more detail on iVector extraction see [14].

### 3.2. Classifier

Once the iVectors for our training data are obtained, a linear generative classifier is trained as proposed in [1]. The distributions of iVectors for individual languages are modeled by Gaussian distributions with a single within-class (WC) full covariance matrix shared by all the languages.

For an iVector $\mathbf{w}$ corresponding to a test utterance, the loglikelihood for each language is

$$\ln p(\mathbf{w}|l) = -\frac{1}{2}\mathbf{w}^T\mathbf{\Sigma}^{-1}\mathbf{w}+\mathbf{w}^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_l-\frac{1}{2}\boldsymbol{\mu}_l^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_l+const,$$

where $\boldsymbol{\mu}_l$ is the mean vector for language $l$, $\mathbf{\Sigma}$ is the common covariance matrix, and $const$ is a language- and iVector-independent constant irrelevant for making decisions. The quadratic term $\mathbf{w}^T\mathbf{\Sigma}^{-1}\mathbf{w}$, which is constant over classes, would be also irrelevant, if the log-likelihoods were directly used to obtain posterior probabilities of classes. However, since the likelihoods are used only as input features to the calibration backend, it makes a difference in our system, as explained in [1].

### 3.3. Fusion and Calibration Backend

For calibration, a Gaussian backend followed by discriminative multiclass logistic regression is used to postprocess scores obtained from the described classifiers. Note that the Gaussian backend is essentially the same model as our generative classifier. However, its inputs are the scores from the classifiers described above rather than the iVectors. Also, it is trained on the separate development dataset to obtain well-calibrated scores. When fusing multiple systems, a separate Gaussian backend is trained for each subsystem and outputs of the Gaussian backends are fused by multiclass logistic regression. A detailed description of the backend, which also uses information about the recording duration for calibration, can be found in [13].

## 4. EXPERIMENTS AND RESULTS

### 4.1. Test Data

Our results are reported for a closed-set task of 3, 10 and 30 seconds of the NIST LRE 2009 evaluation [16]. The data comprises 31178 recordings of 23 target languages. Results are reported in $C_{avg}$, which is an error metric defined in [16].

### 4.2. Training and Development Data

Our training data is from the following databases: CALLFRIEND, NIST LRE03, NIST LRE05, NIST LRE07, and VOA3. The data comprises 51 languages, which are all used to train our UBM. For training iVector extractor matrices T, we use data of only the 23 target languages. For training the generative classifier, we use only 500 files per language, in the same way as in [1].

A separate dataset was used for training the fusion/calibration backend, which includes data from the following databases: CALLFRIEND, CALLHOME, Fisher, NIST LRE05, NIST LRE07, Mixer, OGI22, and VOA.

### 4.3. Results with Prosodic Features

Several parameters can be tuned in the system. We have studied the influence of the number of Gaussians, the iVector dimensionality, and the type of segment definition as described in Section 2.2.

Table 1 compares performance of prosodic features with 1) energy valley based segments and 2) fixed-length segments. UBM with 512 Gaussian components is used in extraction of 300-dimension iVectors. As can be seen, fixed-length segments provide better performance, which is in agreement with the previous experiments on the SID task [10]. Prosodic features with fixed-length segments are used in all the following experiments.

Next, we experimented with the number of Gaussian components in iVector extraction and with iVector dimensionality. Recent experiments in SID [11] show that a reasonable configuration for prosodic systems is 512 Gaussian components and 300-dimension iVectors. Table 2 compares performance of systems with different numbers of Gaussian components. Improvement can be seen when increasing the number of components from 512 to 2048. As for acoustic features [1], increasing the dimensionality of iVectors improves the system accuracy. 400-dimension iVectors were found to be optimal and no additional gains were observed for higher dimensions.

| Condition | Energy valley | Fixed length |
|---|---|---|
| 3 s | 35.08 | **34.57** |
| 10 s | 25.83 | **24.45** |
| 30 s | 19.27 | **17.28** |

**Table 1**. $C_{avg} \times 100$ *on NIST LRE 2009 for the prosodic features with energy valley based segments and fixed-length segments, 512 Gaussian components, 300-dimension iVectors*

| Condition | 512 Gaussians | 1024 Gaussians | 2048 Gaussians |
|---|---|---|---|
| 3 s | 32.56 | 31.97 | **31.76** |
| 10 s | 22.52 | 21.89 | **21.12** |
| 30 s | 15.58 | 14.60 | **13.78** |

**Table 2**. $C_{avg} \times 100$ *on NIST LRE 2009 for the prosodic features with fixed-length segments, 512, 1024 and 2048 Gaussian components, 400-dimension iVectors*

### 4.4. Fusion with Acoustic iVectors-based System

#### *4.4.1. Acoustic system*

The state-of-the-art-acoustic system is built in the same fashion as in [1]. It uses the same configuration (SDC 7-1-3-7, 2048 Gaussians, 600-dimension iVectors) except for not using vocal tract length normalization (VTLN) and having a different training dataset. The UBM, iVector extractor, Gaussian classifier, and backend are trained in the same way and on the same data as described for the prosodic system in Section 4.1. Therefore, the improvements obtained from fusing the acoustic and prosodic system can be attributed to the complementarity of prosodic and cepstral features and not to combining information from different data sources.

#### *4.4.2. Fusion results*

Table 3 shows the results for the state-of-the-art acoustic system, our best prosodic system (2048 Gaussians, 400-dimension iVectors) and the fusion of both systems. As can be seen, the fusion with the prosodic system improves performance in all conditions. The relative improvements obtained over the acoustic system are: 10.93% for 3 seconds; 15.24% for 10 seconds; and 9.39% for 30 seconds.

## 5. CONCLUSIONS

A LID system based on prosodic features has been introduced. Extraction of the pitch, energy, and duration allows us to represent the three components of prosody: stress,

| Condition | Acoustic | Prosodic | Fusion |
|---|---|---|---|
| 3 s | 19.13 | 31.76 | **17.04** |
| 10 s | 6.30 | 21.12 | **5.34** |
| 30 s | 3.09 | 13.78 | **2.80** |

**Table 3**. $C_{avg} \times 100$ *for the generative iVectors-based acoustic system, generative iVectors-based prosodic system and fusion of both systems*

intonation, and rhythm. Unvoiced frames where the pitch is undefined are discarded, permitting us to treat the features as continuous. Thus, the same classifier successfully applied for acoustic LID, based on iVectors and a generative model, can be adapted for our prosodic features. Fixed-length segments, 2048 Gaussians, and 400 dimensions, have been found to be a good configuration for the system. Although the performance of the prosodic system alone does not give outstanding results, it is in the fusion with another LID system where this approach is really powerful. The combination with a prosodic system resulted in significant performance improvements over the state-of-the-art iVectors-based acoustic system on all conditions of the NIST LRE 2009 task. We consider this technique to be very promising as there are still many possibilities for experimenting with additional prosodic features such as AM modulation or formants that could provide further improvements. For this reason, we believe that prosodic features can play an important role in future LID systems. At the same time, a baseline for prosodic systems on the NIST LRE 2009 dataset has been established in this work.

## 6. ACKNOWLEDGMENTS

This work was done during an internship of David Martínez at SRI International funded by the Spanish Ministry of Science and Innovation under project TIN2008-06856-C05-04.

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. D10PC20024. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the Defense Advanced Research Projects Agency (DARPA); or its Contracting Agent, the U.S. Department of the Interior, National Business Center, Acquisition & Property Management Division, Southwest Branch.

## 7. REFERENCES

[1] D. Martínez, O. Plchot, L. Burget, O. Glembek, P. Matějka, “Language identification in iVectors space”, *Proc. Interspeech 2011*, Florence.

[2] N. Brümmer, A. Strasheim, V. Hubeika, P. Matějka, P. Schwarz, J. Černocký, “Discriminative acoustic language recognition via channel-compensated GMM statistics”, *Proc. Interspeech 2009*, Brighton.

[3] M.A. Zissman, “Comparison of four approaches to automatic language identification of telepone speech”, *IEEE Trans. Speech and Audio Proc.*, vol. 4, no. 1, pp. 31-44, 1996.

[4] T. Mikolov, O. Plchot, O. Glembek, P. Matějka, L. Burget, J. Černocký, “PCA-based feature extraction for phonotactic language recognition”, *Proc. Odyssey 2010 - The Speaker and Language Recognition Workshop*, Brno.

[5] M. Soufifar, M. Kockmann, L. Burget, O. Plchot, O. Glembek, “iVector based approach to phonotactic language recognition”, *Proc. Interspeech 2011*, Florence.

[6] L. Mary, B. Yegnanarayana, “Extraction and representation of prosodic features for language and speaker recognition”, *Speech Communication* 50 (2008) p. 782-796.

[7] Chi-Yueh Lin, Hsiao-Chuan Wang, “Language identification using pitch contour information”, *Proc. ICASSP 2005*, Philadelphia.

[8] N. Dehak, P. Demouchel, P. Kenny, “Modeling prosodic features with joint factor analysis for speaker verification”, *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 2095-2103, Sept. 2007.

[9] L.Ferrer, N. Scheffer, E. Shriberg, “A comparison of approaches for modeling prosodic features in speaker recognition”, *Proc. ICASSP 2010*, Dallas.

[10] M. Kockmann, L. Burget, J. Černocký, “Investigations into prosodic syllable contour features for speaker recognition”, *Proc. ICASSP 2010*, Dallas.

[11] M. Kockmann, L. Ferrer, L. Burget, and J. H. Cernock, ”iVector fusion of prosodic and cepstral features for speaker verification”, *Proc. Interspeech 2011*, Florence.

[12] M. Kockmann, L. Burget, O. Glembek, L. Ferrer, J. Černocký, “Prosodic speaker verification using subspace multinomial models with intersession compensation”, *Proc. Interspeech 2010*, Makuhari.

[13] Z. Jančík, O. Plchot, N. Brümmer, L. Burget, O. Glembek, V. Hubeika, M. Karafiát, P. Matějka, T. Mikolov, A. Strasheim, J. Černocký, “Data selection and calibration issues in automatic language recognition - investigation with BUT-AGNITIO NIST LRE 2009 system”, *Proc. Odyssey 2010 - The Speaker and Language Recognition Workshop*, Brno.

[14] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel and P. Ouellet, “Front-end factor analysis for speaker verification”, *IEEE Trans. on Audio, Speech and Language Processing*, vol. 19, pp. 788-798, May 2011.

[15] K. Sjölander, “The Snack Sound Toolkit”, http://www.speech.kth.se/snack/.

[16] “The 2009 NIST Language Recognition Evaluation Plan (LRE09)”, http://www.itl.nist.gov/iad/mig/tests/lre/2009/LRE09_EvalPlan_v6.pdf.

# iVector-Based Discriminative Adaptation for Automatic Speech Recognition

Martin Karafiát #1, Lukáš Burget *#2, Pavel Matějka #3, Ondřej Glembek #4, Jan "Honza" Černocký #5,

# *Brno University of Technology, Speech@FIT, Božetěchova 2, Brno, 612 66, Czech Republic*
[1] karafiat@fit.vutbr.cz, [3] matejkap@fit.vutbr.cz, [4] glembek@fit.vutbr.cz [5] cernocky@fit.vutbr.cz

* *SRI International, 333 Ravenswood Avenue, Menlo Park, 94025, CA, USA*
[2] burget@speech.sri.com

***Abstract*—We presented a novel technique for discriminative feature-level adaptation of automatic speech recognition system. The concept of iVectors popular in Speaker Recognition is used to extract information about speaker or acoustic environment from speech segment. iVector is a low-dimensional fixed-length representing such information. To utilized iVectors for adaptation, Region Dependent Linear Transforms (RDLT) are discriminatively trained using MPE criterion on large amount of annotated data to extract the relevant information from iVectors and to compensate speech feature. The approach was tested on standard CTS data. We found it to be complementary to common adaptation techniques. On a well tuned RDLT system with standard CMLLR adaptation we reached 0.8% additive absolute WER improvement.**

## I. Introduction

We propose new method for discriminative adaptation of automatic speech recognition (ASR) system, which is based on combination of two successful techniques: From speaker recognition field, we have borrowed the idea of representing speech segment using so called iVector. iVector is an information-rich low-dimensional fixed length vector extracted from the feature sequence. Recently, systems based on iVectors [1], [2], [3] extracted from cepstral features have provided excellent performance in speaker verification, which classifies iVectors as good candidates for representing information about speaker. Just like MLLR transformations for ASR adaptation became popular features in speaker recognition [4], we believe that iVectors — successful in speaker recognition — can be used as compact representations for ASR adaptation. For brevity, we will describe the proposed method only from the perspective of speaker adaptation. Keep in mind, however, that iVector represents information about both speaker and acoustic environment of the corresponding segment and therefore, the proposed technique is expected to effectively adapt ASR system to both speaker and acoustic environment.

In order to utilize information encoded in iVectors for adaptation of speech recognition system, we build on the idea of Region Dependent Linear Transform (RDLT) [5]. In the original version, RDLT is a nonlinear feature transformation, which is typically discriminatively trained using Minimum Phone Error (MPE) criterion [6]. More precisely, each feature vector is transformed by a linear transformation, which is selected from an ensemble of transformations depending on the acoustic region of the current frame. To apply this framework for discriminatively trained feature-level adaptation, we use the same form of frame-dependent transformation. However, the *fixed* iVector is transformed by such varying transformation and the resulting vector is added as a bias to the original feature vector.

The paper is organized as follows: the following section II presents the state-of-the-art in discriminative techniques for speaker adaptation and positions our proposal. Section III briefly introduces iVectors while IV defines the RDLT scheme and recipes. Section V suggests the the iVector adaptation. The following section VI describes the experimental setup including the baseline systems and VII presents the results with RDLT systems including the proposed iVector adaptation. Section VIII contains the conclusions and directions for future work.

## II. Current techniques for discriminative adaptation and position of our proposal

The idea of using discriminative training criterion for adaptation is not new. In the early works on this topic [7], [8], [9], acoustic model parameters or features were adapted using transformations of the same form as in Maximum Likelihood Linear Regression (MLLR) or Constrained MLLR (CMLLR), where the adaptation transformations were estimated on adaptation data by optimizing discriminative rather than Maximum Likelihood (ML) criterion. While this approach provided excellent performance for supervised adaptation, it appeared to be too sensitive to the quality of the initial hypothesis in the case of unsupervised adaptation. Fortunately, our technique does not suffer form such problem as only the RDLT part is trained using MPE criterion on large amount of annotated training data. RDLT discriminatively adapts speech features based on the information encoded in the iVector. The iVectors estimated on adaptation data are, however, robustly obtained by optimizing Maximum a-posteriori (MAP) criterion. Moreover, there is no need for any initial hypothesis as iVectors are estimated using simple Gaussian Mixture Model (GMM).

Our technique is similar in spirit to Discriminative Mapping transforms (DMT) [10], [11], where MLLR or CMLLR transformations are estimated on adaptation data using ML criterion first. The adapted model parameters are further post-processed by an ensemble of discriminatively trained linear transformations (typically 64), where each transformation corresponds to a cluster of Gaussian components from the acoustic model. The transformations are discriminatively trained on large amount of annotated training data to refine the adapted models and to compensate for the discriminative power that could be taken away from discriminatively acoustic trained models when adapted using ML estimated transformations.

DMT can be seen as some form of region dependent transforms, where the regions in acoustic space are defined by the Gaussian clusters rather than by a dedicated GMM as it is in the case of RDLT. From this perspective, CMLLR-based DMT [11] is very similar to standard RDLT jointly trained with the following CMLLR adaptation as described in [5]. Therefore, it can be expected that, just like RDLT, DMT would bring improvements even without ML trained adaptation transformations. Unfortunately, the papers on DMT do not provide such analysis and it is not clear how much improvement is to be attributed to improved adaptation and how much to the improved discriminative acoustic model training.

In our approach, however, we do not estimate any feature or model transformations to adapt the acoustic model to the adaptation data. Instead, we estimate iVector summarizing information about the speaker and the acoustic environment of adaptation data independently of any ASR acoustic model. Also, the discriminatively trained transformation does no directly operate on speech features or model parameters. Instead, for each speech frame, it is trained to extract a correction bias vector from iVector. In our implementation, zero iVector, which is the expected value of iVector on training data, leads to zero correction bias and therefore to no adaptation. Therefore, it is easy to separately analyze the effect of RDLT used for adaptation and RDLT used, in the standard way, as a discriminative feature transformation.

## III. iVECTORS

The iVector approach has become state of the art in the speaker verification field [1]. In this work, we show that it can be successfully applied to extract information useful for adapting ASR system. The approach provides an elegant way of reducing large-dimensional sequential input data to a low-dimensional fixed length feature vector while retaining most of the relevant information.

In the iVector framework, a GMM model is adapted to observation sequence representing a speech segment that we want to extract speaker information from. Only the mean parameters of a pre-trained GMM are adapted. The supervector of concatenated mean vectors for the adapted GMM is obtained as

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{i}, \qquad (1)$$

where $\mathbf{m}$ is the segment-independent component of the mean supervector, $\mathbf{T}$ is a matrix of basis spanning the subspace covering the important variability (both useful and useless for adaptation) in the supervector space, and $\mathbf{i}$ is a low-dimensional latent variable representing coordinates in the subspace. We assume standard normal prior for the latent variable $\mathbf{i}$. GMM is adapted to the observation sequence by finding $\mathbf{i}$ that maximizes MAP criterion. This MAP point estimate of $\mathbf{i}$, which is obtained with single iteration of EM algorithm, is taken as the iVector representing the segment. The parameters of the GMM and the subspace are trained in unsupervised manner using EM algorithm on a collection of speech segment covering variety of speakers and acoustic environments. We use an efficient implementation of the training procedure suggested in [12].

## IV. REGION DEPENDENT LINEAR TRANSFORMS

In the RDLT framework, an ensemble of linear transformations is trained discriminatively. Each transformation corresponds to one region in partitioned feature space. Each feature vector is then transformed by a linear transformation corresponding to the region that the vector belongs to. The resulting (generally nonlinear) transformation has the following form:

$$F_{RDLT}(\mathbf{o}_t) = \sum_{r=1}^{N} \gamma_r(t)(\mathbf{A}_r\mathbf{o}_t + \mathbf{b}_r), \qquad (2)$$

where $o_t$ is input feature vector at time $t$, $\mathbf{A}_r$ and $\mathbf{b}_r$ are linear transformation and biases corresponding $r$th region and $\gamma_r(t)$ is probability that the vector $\mathbf{o}_t$ belongs to $r$th region. The probabilities $\gamma_r(t)$ are typically obtained using GMM (pre-trained on the input features) as mixture component posterior probabilities. Usually, RDLT parameters $\mathbf{A}_r$, $\mathbf{b}_r$ and ASR acoustic model parameters are alternately updated in several iterations. While RDLT parameters are updated using MPE criterion, ML update is typically used for acoustic model parameters. As proposed in [13] and described in RDLT context in [5], ML update of acoustic model parameters must be taken into account when optimizing RDLT parameters. Otherwise, the discriminative power obtained from MPE training of RDLT feature transformation is mostly lost after ML acoustic model re-training. In our experiments, we closely follow the training recipe described in [5].

In our experiments, we do not use the bias terms $\mathbf{b}_r$ (the number of their parameters would anyway be only a small proportion of parameters in matrices $\mathbf{A}_r$). In agreement with results reported in [5], we have found that omitting the bias terms has little effect on the performance.

RDLT can be seen as a generalization of previously proposed fMPE discriminative feature transformation. The special case of RDLT with square matrices $\mathbf{A}_r$ (i.e. without dimensionality reduction of input features) was shown [5] to be equivalent to fMPE with offset features as described in [14]. This is also the configuration used in our experiments. From fMPE recipe [13], we have also take the idea of incorporating context information by considering $\gamma_r(t)$ corresponding not only to the current frame but also to the neighboring frames (see section VII-A for more details). From our experience, this style of incorporation context information leads to significantly better results compared to to the style previously considered in the context of RDLT [5], where feature vectors of multiple frames were stacked at the RDLT input and transformations with dimensionality reduction were used to recover the original feature dimensionality. Therefore, our RDLT baseline system configuration is very similar to the one described in the fMPE recipe. Still, we prefer to use the more general RDLT abstraction as it can be easily extended by the proposed iVector based adaptation.

## V. iVECTOR BASED ADAPTATION

To utilize the RDLT framework for adaptation, we use transformation of the following form:

$$F_{ivec}(\mathbf{o}_t) = \mathbf{o}_t + \sum_{r=1}^{N} \gamma_r(t)\mathbf{A}_r\mathbf{i}_s, \tag{3}$$

where $\mathbf{i}_s$ is iVector estimated on adaptation data corresponding to speaker $s$. Typically, iVector dimensionality is larger than the dimensionality of feature vector, therefore $\mathbf{A}_r$ are matrices reducing the dimensionality of iVector to the one of feature vectors. The same MPE training framework as described in the previous section can be used to train RDLT to discriminatively extract the corrective term from iVector $\mathbf{i}_s$, which is added to the original feature vector $\mathbf{o}_t$ in order to adapt the features to the model. Note that, although the iVector stays constant, its transformation depends or region of current feature frame so that different pieces of information can be extracted from iVector to compensate feature frames from different regions of acoustic space.

We again use the iterative training scheme where, after updating RDLT parameters, acoustic model parameters are retrained on the compensated features. The resulting procedure can be seen as another form of speaker adaptive training (SAT) [15], [16].

Finally, we can combine both ideas of using RDLT for adaptation and discriminative feature transformation. Since the whole RDLT framework has to be implemented to deal with either of the two problems, it makes a little sense to use RDLT only for adaptation without using it also for feature transformation, which is expected to provide an additional significant gain. If the same data and the same region definitions are used to train RDLT for both problems, which is the case in our experiments, we can simply concatenate each feature vector with the appropriate iVector and process the resulting extended vectors

$$\tilde{\mathbf{o}}_t = \begin{bmatrix} \mathbf{o}_t \\ \mathbf{i}_s \end{bmatrix} \tag{4}$$

just as in the standard RDLT framework corresponding to equation (2). $\mathbf{A}_r$ will perform dimensionality reduction.

## VI. EXPERIMENTAL SETUP

### A. ASR training and testing data

The acoustic model was trained on ctstrain04 training set, a subset of the h5train03 set, defined at the University of Cambridge. It contains about 278 hours of well transcribed speech data from Switchboard I,II and Call Home English (see Table I).

| Database | Amount of data [hours] |
|---|---|
| Switchboard I | 248.52 |
| Switchboard cellular | 15.27 |
| Call Home English | 13.93 |
| Total | 277.72 |

TABLE I
CTS TRAINING DATA DESCRIPTION.

| Models | WER [%] |
|---|---|
| ML | 34.7 |
| ML - CMLLR | 32.1 |
| ML - CMLLR-SAT | 31.9 |

TABLE II
BASELINE: ML TRAINED SYSTEMS

All recognition results are reported on the Hub5 Eval01 test set (defined during 2001 NIST CTS evaluation) composed of 3 subsets of 20 conversations from Switchboard-1, Switchboard-2 and Switchboard-cellular corpora, for a total length of more than 6 hours of audio data.

A bigram language model was used for recognition. It was adopted from AMI speech recognition system for NIST Rich Transcriptions 2007 [17].

### B. Baseline ASR systems

The speech recognition system is HMM-based cross-word tied-states triphones, with approximately 8500 tied states and 28 Gaussian mixtures per state. The features were 13 VTLN normalized Mel-Frequency PLP coefficients generated by HTK, augmented with their deltas, double-deltas and triple-deltas. Cepstral mean and variance normalization was applied with the mean and variance vectors estimated on each conversation side. HLDA was estimated with Gaussian components as classes and the dimensionality was reduced from 52 to 39. This model is denoted as ML in table II

Using this model, CMLLR adaptation transforms were generated for training and test data, one for each conversation side. This model also served for generating lattices, which were used for MPE training of RDLT. Only a single CMLLR transformation was used in our system, as we did not observe any significant gain from using multiple CMLLR or MLLR transformations with our system on this task. Table II shows 2.6% absolute improvement in Word Error Rate (WER)obtained from CMLLR adaptation and additional 0.2% WER improvement when the acoustic model was retrained in SAT fashion [16]. Unless stated otherwise, CMLLR SAT system forms the basis of all systems described in the following sections.

### C. iVector extraction

In principle, both ASR acoustic models and iVector extraction could be based on the same features and trained on the same data. Also, iVector extraction and definition of regions in RDLT could be based on the same GMM model. In our experiments, however, we use two different GMMs trained on different features, since we simply took iVectors extracted by our existing system optimized for speaker verification task [3].

The features used for the iVector extraction were 19 Mel frequency cepstral coefficients (with log-energy) calculated every 10 ms using 25 ms Hamming window. This 20-dimensional feature vector was subjected to short time mean and variance normalization using a 3s sliding window. Delta and double delta coefficients were then calculated using a 5-frame window giving 60-dimensional feature vectors. The iVector extraction was based on Semi-Tied Covariance (STC) GMM with 2048 mixture components, which was trained on NIST SRE 2004 and 2005 telephone data. The subspace matrix $\mathbf{T}$ was trained on more than 2500 hours of data from the following telephone databases: NIST SRE 2004, 2005, 2006, Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, Fisher English Parts 1 and 2. The results are reported with 400 dimensional iVectors. Similarly to CMLLR transformations, iVectors were generated per conversation side for training and test data.

One could object that the iVector extraction is trained on much more data than the baseline ASR system, which makes the comparison of systems unfair. However, the iVector extraction is trained in *unsupervised manner* on data that are mostly not transcribed and therefore unusable for ASR training. Also, while large amount of training data is necessary to obtain good performance is speaker verification, we believe that it is not the case in these experiments, as RDLT, which is trained to extract the adaptation information from the iVector, is still trained on the same data as baseline ASR system.

## VII. RDLT EXPERIMENTS

### A. RDLT for discriminative feature extraction

In this section, we examine different configurations of RDLT used only in the usual way as a discriminative feature extraction. In the trivial case, where all feature frames are considers to belong to only one single region, RDLT comprises only one discriminatively trained linear transform. This configuration, which is also know as Discriminative HLDA [18], brings 0.5% absolute WER improvement compared to "ML CMLLR-SAT" baseline, as we can see in the first line of Table III.

The second line of the table reports additional 1.1% absolute WER improvement obtained from using 1000 regions. To define the regions in the acoustic space, all Gaussians from ML trained HMM model are pooled and clustered using agglomerative clustering to create GMM with desired number of components (see [19] for detailed description of the clustering algorithm).

In the following experiment, we incorporated also the information about context by using region posterior probabilities also from neighboring frames as suggested in [13]. Posterior probabilities of the GMM components for a current frame are stacked with the averages of posteriors for adjacent frames 1-2, 3-5 and 6-9 on the right and likewise for the left context (i.e. 7 groups spanning 19 frames in total). The resulting 7000 dimensional vector served as weights $\gamma_r(t)$ in equation (2) corresponding to 7000 transformations (39×39 matrices). Block diagram demonstrating such RDLT configuration is shown in Figure 1. The use of context brings large additional improvement (3% absolute) as can be seen in Table III in line denoted as "RDLT 7x1000 regions".

| Models | WER [%] |
|---|---|
| RDLT 1 regions | 31.4 |
| RDLT 1000 regions | 30.3 |
| RDLT 7x1000 regions | **27.3** |
| RDLT 7x500 regions | 27.6 |
| RDLT 7x250 regions | 27.7 |

TABLE III
RESULTS WITH RDLT USED AS FEATURE TRANSFORMATION FOR CMLLR-SAT ADAPTED SYSTEM.

| Models | WER [%] |
|---|---|
| iVector RDLT 1 region | 31.3 |
| iVector RDLT 250 regions | 30.2 |
| iVector RDLT 500 regions | 30.0 |
| iVector RDLT 1000 regions | 29.9 |

TABLE IV
RESULTS WITH RDLT USED ONLY FOR IVECTORS BASED ADAPTATION APPLIED ON TOP OF CMLLR-SAT ADAPTATION.

Next, we tested scaled-down systems to see a degradation of performance with smaller number of regions. A difference in WER between 1000 and 250 regions is 0.4%. This suggests that it is more important to invest parameters into context modeling than increasing the number of regions for the current frame.

### B. iVector based adaptation

Table IV shows the behavior of the proposed adaptation approach with various number of transforms. To find the optimal configuration, we first considered the case corresponding to equation (3), where RDLT is used only for the adaptation. The optimal number of transformations saturates again on 1000 giving 2% absolute WER improvement over the CMLLR-SAT baseline. The differences between 500 and 1000 mixture components (and hence regions) is only 0.1% absolute.

We also experimented with incorporating the context information using the region posteriors form neighboring frames, but we found it ineffective when using RDLT for adaptation.

In table V, we compare the effect of CMLLR adaptation, iVector adaptation and combination of both for systems with and without RDLT used as discriminative feature transformation. For RDLT as feature transformation, we use the configuration with 7000 transformations as described in the previous section. For iVector adaptation, RDLT uses only 1000 transformations corresponding only to the regions for the current frame. This is the case even when both RDLT for feature transformation and RDLT for adaptation are combined. In this case, only 1000 transformations (39×439 matrices) corresponding to the current frame of GMM posteriors pro-

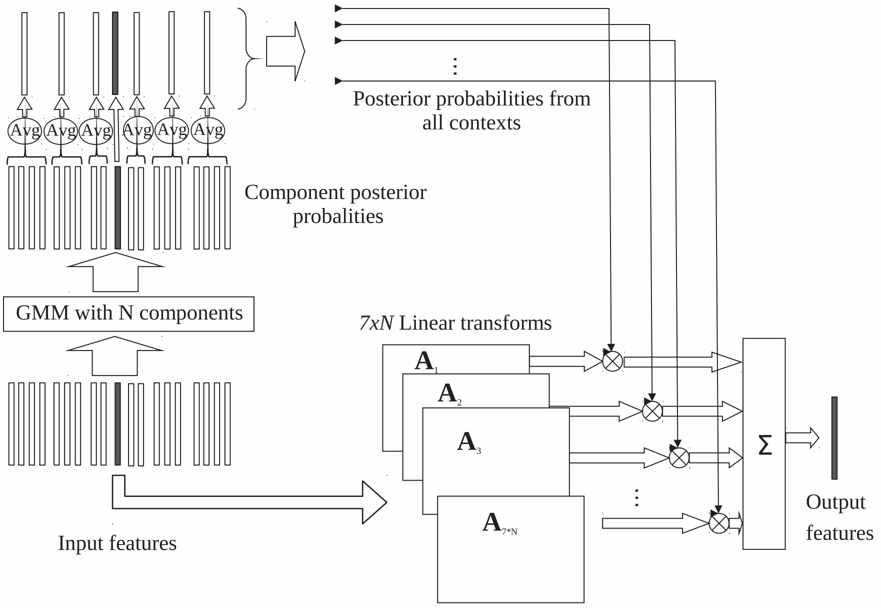


Fig. 1. RDLT with context transformations.

| Adaptation | ML | RDLT |
|---|---|---|
| none | 34.7 | 29.7 |
| iVector | 32.1 | 28.7 |
| CMLLR-SAT | 31.9 | 27.3 |
| both | 29.9 | 26.5 |

TABLE V
SUMMARY OF DIFFERENT TECHNIQUES.

cesses 39-dimensional feature vector concatenated with 400 dimensional iVector. The remaining transformations (39×39 matrices) corresponding to context posteriors process only the 39-dimensional feature vector.

The first line of table V shows the results without any adaptation. As can be seen, RDLT provides impressive improvement 5% absolute in this case. Comparing the following two lines, we see that iVector adaptation on its own appears to be slightly less effective than CMLLR transformation for this task. However, the two adaptation techniques seem to be complementary and the best result is obtained from their combination as can be seen from the last line in the table.

## VIII. CONCLUSIONS AND FUTURE WORK

We presented a novel technique for feature compensation based on iVectors — a popular technique in Speaker Recognition. We found it to be complementary approach to common adaptation techniques. On a well tuned RDLT system with standard CMLLR adaptation, we reached 0.8% additive absolute WER improvement. Without CMLLR adaptation, 1.0% absolute improvement was obtained.

Unsupervised estimation of CMLLR requires an additional decoding pass to obtain the adaptation hypothesis. On contrary, our approach only requires to extract the iVector from adaptation data which takes only a fraction of time necessary for decoding. Forwarding features through the set of transforms is also fast as only few transformations (usually only one or two) are applied per frame due to the sparsity of posterior probabilities. Therefore, our approach could be considered for decoding in the first pass of multi-pass systems or in one-pass systems.

This paper presents the first results of the proposed technique, in short-term, we will face the following issues:

1) Lattices used for discriminative training were generated using model with more than 8% higher WER compared to the performance of the final model. Further improvement could be obtain from lattices that would better reflect errors made by the final system.
2) iVector extraction was optimized for Speaker Recognition and the optimal configuration for speech recognition can be very different. Also, iVector extraction based on ASR features and GMM taken from RDLT would greatly simplify the system.
3) Finally, we would like to integrate the proposed approach into our full-featured system including other advanced techniques such as MPE model parameter training or neural network bottle-neck features.

This paper describe only one special instance of a more general scheme, where nonlinear transformation is trained discriminatively to compensate features based on external source of information useful for adaptation. Other forms of discriminatively trained nonlinear transformations can be considered (e.g. artificial neural networks), and different external sources of adaptation information can be found useful (e.g. noise spectrum estimate for noise robust speech recognition).

## Acknowledgment

This work was partly supported by Technology Agency of the Czech Republic grant No. TA01011328, Czech Ministry of Education project No. MSM0021630528, Grant Agency of Czech Republic project No. 102/08/0707, and by Czech Ministry of Trade and Commerce project No. FR-TI1/034.

## References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front–end factor analysis for speaker verification," in *IEEE Trans. on Audio, Speech and Lang. Process.*, 2010.

[2] P. Kenny, "Bayesian speaker verification with heavy–tailed priors," keynote presentation, Proc. of Odyssey 2010, Brno, Czech Republic, June 2010.

[3] P. Matějka, O. Glembek, F. Castaldo, J. Alam, O. Plchot, P. Kenny, L. Burget, and J. Černocký, "Full-covariance UBM and heavy-tailed PLDA in I-vector speaker verification," in *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*. IEEE Signal Processing Society, 2011, pp. 4828–4831.

[4] A. Stolcke, L. Ferrer, and S. Kajarekar, "Improvements in mllrtransform-based speaker recognition," in *in Proc. IEEE Odyssey 2006 Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006.

[5] B. Zhang, S. Matsoukas, and R. Schwartz, "Recent progress on the discriminative region-dependent transform for speech feature extraction," in *Proc. of Interspeech2006*, Pittsburgh, PA, USA, Sep 2006, pp. 2977–2980.

[6] D. Povey, "Discriminative training for large vocabulary speech recognition," Ph.D. dissertation, University of Cambridge, 2003.

[7] A. Gunawardana and W. Byrne, "Discriminative speaker adaptation with conditional maximum likelihood linear regression," in *Proc. Eurospeech 2001*, Aalborg, Denmark, Sep. 2001.

[8] L. F. Uebel and P. C.Woodland, "Discriminative linear transforms for speaker adaptation," in *Proc. ISCA ITRW on Adaptation Methods for Speech Recognition*, 2001.

[9] S. Tsakalidis, V. Doumpiotis, and W. J. Byrne, "Discriminative linear transforms for feature normalization and speaker adaptation in hmm estimation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 13, pp. 367–376, 2005.

[10] K. Yu, M. J. F. Gales, and P. C. Woodland, "Unsupervised discriminative adaptation using discriminative mapping transforms," in *Proc. ICASSP, LAS VEGAS, NV*, Las Vegas, NV, USA, 2008, pp. 4273–4276.

[11] L. Chen, M. J. F. Gales, and K. K. Chin, "Constrained discriminative mapping transforms for unsupervised speaker adaptation," in *Proc. ICASSP*, Prague, Czech Republic, 2008.

[12] O. Glembek, L. Burget, P. Kenny, M. Karafiát, and P. Matějka, "Simplification and optimization of i-vector extraction," in *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*. IEEE Signal Processing Society, 2011, pp. 4516–4519.

[13] D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, and G. Zweig, "fmpe: Discriminatively trained features for speech recognition," in *in Proc. IEEE ICASSP*, 2005.

[14] D. Povey, "Improvements to fMPE for discriminative training of features," in *Proc. of Interspeech2005*, Lisbon, Portugal, Sep 2005, pp. 2977–2980.

[15] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training," in *Proc. ICSLP '96*, vol. 2, Philadelphia, PA, 1996, pp. 1137–1140. [Online]. Available: citeseer.ist.psu.edu/anastasakos96compact.html

[16] M. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," 1997. [Online]. Available: citeseer.ist.psu.edu/gales97maximum.html

[17] T. Hain *et al.*, "The 2007 AMI(DA) system for meeting transcription," in *Proc. Rich Transcription 2007 Spring Meeting Recognition Evaluation Workshop*, Baltimore, Maryland USA, May 2007.

[18] B. Zhang and S. Matsoukas, "Minimum phoneme error based heteroscedastic linear discriminant analysis for speech recognition," in *Proc. IEEE Int. Conf. on Acoustic, Speech and Signal processing (ICASSP)*, Philadelphia, PA, USA, march 2005, pp. 925–929.

[19] D. Povey, L. Burget, M. Agarwal, P. Akyazi, A. Ghoshal, O. Glembek, K. N. Goel, M. Karafiát, A. Rastrow, R. Rose, P. Schwarz, and S. Thomas, "The subspace gaussian mixture model-a structured model for speech recognition," *Computer Speech and Language*, vol. 25, no. 2, pp. 404–439, 2011.

# DISCRIMINATIVELY TRAINED PROBABILISTIC LINEAR DISCRIMINANT ANALYSIS FOR SPEAKER VERIFICATION

Lukáš Burget[1], Oldřich Plchot[1], Sandro Cumani[2], Ondřej Glembek[1], Pavel Matějka[1], Niko Brümmer[3]

[1]Brno University of Technology, Czech Rep., {burget,iplchot,glembek,matejkap}@fit.vutbr.cz,
[2]Politecnico di Torino, Italy, sandro.cumani@polito.it, [3]AGNITIO, S. Africa, niko.brummer@gmail.com

## ABSTRACT

Recently, i-vector extraction and Probabilistic Linear Discriminant Analysis (PLDA) have proven to provide state-of-the-art speaker verification performance. In this paper, the speaker verification score for a pair of i-vectors representing a trial is computed with a functional form derived from the successful PLDA generative model. In our case, however, parameters of this function are estimated based on a discriminative training criterion. We propose to use the objective function to directly address the task in speaker verification: discrimination between same-speaker and different-speaker trials. Compared with a baseline which uses a generatively trained PLDA model, discriminative training provides up to 40% relative improvement on the NIST SRE 2010 evaluation task.

***Index Terms—*** Speaker verification, Discriminative training, Probabilistic Linear Discriminant Analysis

## 1. INTRODUCTION

In this paper, we show that discriminative training can be used to improve the performance of state-of-the-art speaker verification systems based on i-vector extraction and Probabilistic Linear Discriminant Analysis (PLDA). Recently, systems based on i-vectors [1, 2] extracted from cepstral features have provided superior performance in speaker verification. The so-called i-vector is an information-rich low-dimensional fixed length vector extracted from the feature sequence representing a speech segment (see section 2 for more details on i-vector extraction). A speaker verification score is then produced by comparing the two i-vectors corresponding to the segments in the verification trial. The function taking two i-vectors as an input and producing the corresponding verification score is typically designed to give a good approximation of the log-likelihood ratio between the "same-speaker" and "different-speaker" hypotheses. Typically, the function is also designed to produce a symmetric score (i.e. to produce output that is independent of which segment is enrollment and which is test — unlike traditional systems, which distinguish the two). In [1], good performance was reported when scores were computed as cosine distances between i-vectors normalized using within-class covariance normalization (WCCN). Best performance, however, is currently obtained with PLDA [2] — a generative model that models i-vector distributions allowing for direct evaluation of the desired log-likelihood ratio verification score (see section 3 for details on the specific form of PLDA used in our work).

In this paper, we propose to estimate verification scores using a *discriminative model* rather than a generative PLDA model. More specifically, the speaker verification score for a pair of i-vectors is computed using a function having the functional form derived from the PLDA generative model. The parameters of the function, however, are estimated using a discriminative training criterion. We use an objective function that directly addresses the speaker verification task, i.e. the discrimination between "same-speaker" and "different-speaker" trials. In other words, a binary classifier that takes a pair of i-vectors as an input, is trained to answer the question of whether or not the two i-vectors come from the same speaker. We show that the functional form derived from PLDA can be interpreted as a binary linear classifier in a nonlinearly expanded space of i-vector pairs. We have experimented with two discriminative linear classifiers, namely linear support vector machines (SVM) and logistic regression. The advantage of logistic regression is its probabilistic interpretation: the linear output of this classifier can be directly interpreted as the desired log-likelihood ratio verification score. On the NIST SRE 2010 evaluation task, we show that up to 40% relative improvement over the PLDA baseline can by obtained with such discriminatively trained models.

There has been previous work on discriminative training for speaker recognition, such as GMM-SVM [3]. This and similar approaches, however, do not directly address the objective of discriminating between same-speaker and different-speaker trials. Instead, SVMs are trained as discriminative models representing each target speaker. As a consequence, this approach cannot fully benefit from discriminative training, as there is a very limited number of positive examples (usually only one enrollment segment) available for training of each model. In contrast, in our approach, a model is trained using a large number of positive and negative examples, each of which is one of many possible same-speaker or different-speaker trials that can be constructed from the training segments.

The very same idea of discriminatively training a PLDA-like model for speaker verification was originally proposed in [4] and some initial work has been done in [5]. At that time, however, speaker factors extracted using Joint Factor Analysis (JFA) [6] were used as a suboptimal input for the classifier, and state-of-the-art performance would not have been achieved.

## 2. I-VECTORS

The i-vector approach has become state of the art in the speaker verification field [1]. The approach provides an elegant way of reducing large-dimensional input data to a small-dimensional feature vector while retaining most of the relevant information. The technique was originally inspired by the JFA framework [6]. The basic principle is that on some data, we train the i-vector extractor and then for

This work was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory (ARL). All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI, or the U.S. Government. The work was also partly supported by the Grant Agency of Czech Republic project No. 102/08/0707, and Czech Ministry of Education project No. MSM0021630528.

each speech segment, we extract the i-vector as a low-dimensional fixed length representation of the segment. The main idea is that the speaker- and session-dependent supervectors of concatenated Gaussian mixture model (GMM) means can be modeled as

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{x}, \tag{1}$$

where $\mathbf{m}$ is the Universal Backgroung Model (UBM) GMM mean supervector, $\mathbf{T}$ is a matrix of bases spanning the subspace covering the important variability (both speaker- and session-specific) in the supervector space, and $\mathbf{x}$ is a standard-normally distributed latent variable. For each observation sequence representing a segment, our i-vector $\boldsymbol{\phi}$ is the MAP point estimate of the latent variable $\mathbf{x}$.

## 3. PLDA

### 3.1. Two covariance model

To facilitate comparison of i-vectors in a verification trial, we model the distribution of i-vectors using a Probabilistic LDA model [7, 2]. We first consider only a special form of PLDA, a *two-covariance model*, in which speaker and inter-session variability are modeled using across-class and within-class full covariance matrices $\boldsymbol{\Sigma}_{ac}$ and $\boldsymbol{\Sigma}_{wc}$. The two-covariance model is a generative linear-Gaussian model, where latent vectors $\mathbf{y}$ representing speakers (or more generally classes) are assumed to be distributed according to prior distribution

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma}_{ac}). \tag{2}$$

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

$$p(\boldsymbol{\phi}|\hat{\mathbf{y}}) = \mathcal{N}(\boldsymbol{\phi}; \hat{\mathbf{y}}, \boldsymbol{\Sigma}_{wc}). \tag{3}$$

The ML estimates of the model parameters, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}_{ac}$, and $\boldsymbol{\Sigma}_{wc}$, can be obtained using an EM algorithm as in [2]. The training i-vectors come from a database comprising recordings of many speakers (to capture across-class variability), each recorded in several sessions (to capture within-class variability).

In the more general case, the speaker and/or inter-session variability can be modeled using subspaces [1]. For example, in our baseline system, speaker variability is not modeled using a full covariance matrix. Instead a low rank across-class covariance matrix is modeled as $\boldsymbol{\Sigma}_{ac} = \mathbf{V}^T\mathbf{V}$, which limits speaker variability to live in a subspace spanned by the columns of the reduced rank matrix $V$.

### 3.2. Evaluation of verification score

Consider the process of generating two i-vectors $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ forming a trial. In the case of a same-speaker trial, a single vector $\hat{\mathbf{y}}$ representing a speaker is generated from the prior $p(\mathbf{y})$, for which both $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ are generated from $p(\boldsymbol{\phi}|\hat{\mathbf{y}})$. For a different-speaker trial, two latent vectors representing two different speakers are independently generated from $p(\mathbf{y})$. For each latent vector, one of the i-vectors $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ is generated. Given a trial, we want to test two hypotheses: $\mathcal{H}_d$ that the trial is a different-speaker trial and $\mathcal{H}_s$ that the trial is a same-speaker trial. The speaker verification score can now be calculated as a log-likelihood ratio between the two hypotheses $\mathcal{H}_s$ and $\mathcal{H}_d$ as

$$\begin{aligned} s &= \log \frac{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|\mathcal{H}_s)}{p(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|\mathcal{H}_d)} && (4) \\ &= \log \frac{\int p(\boldsymbol{\phi}_1|\mathbf{y})p(\boldsymbol{\phi}_2|\mathbf{y})p(\mathbf{y})d\mathbf{y}}{p(\boldsymbol{\phi}_1)p(\boldsymbol{\phi}_2)}, && (5) \end{aligned}$$

where in the numerator we integrate over the distribution of speaker vectors and, for each possible speaker, the likelihood of producing both i-vectors from the speaker is calculated. In the denominator, we simply multiply the marginal likelihoods $p(\boldsymbol{\phi}) = \int p(\boldsymbol{\phi}|\mathbf{y})p(\mathbf{y})d\mathbf{y}$. The integrals, which can be interpreted as convolutions of Gaussians, can be evaluated analytically giving

$$\begin{aligned} s &= \log \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right) \\ &- \log \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right), \qquad (6) \end{aligned}$$

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

$$\begin{aligned} s &= \boldsymbol{\phi}_1^T\boldsymbol{\Lambda}\boldsymbol{\phi}_2 + \boldsymbol{\phi}_2^T\boldsymbol{\Lambda}\boldsymbol{\phi}_1 + \boldsymbol{\phi}_1^T\boldsymbol{\Gamma}\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2^T\boldsymbol{\Gamma}\boldsymbol{\phi}_2 \\ &+ (\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2)^T\mathbf{c} + k, \qquad (7) \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{\Gamma} &= -\frac{1}{4}(\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1} - \frac{1}{4}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{2}\boldsymbol{\Sigma}_{tot}^{-1} \\ \boldsymbol{\Lambda} &= -\frac{1}{4}(\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1} + \frac{1}{4}\boldsymbol{\Sigma}_{wc}^{-1} \\ \mathbf{c} &= ((\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1} - \boldsymbol{\Sigma}_{tot}^{-1})\boldsymbol{\mu} \\ k &= \log|\boldsymbol{\Sigma}_{tot}| - \frac{1}{2}\log|\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac}| - \frac{1}{2}\log|\boldsymbol{\Sigma}_{wc}| \\ &\quad +\boldsymbol{\mu}^T(\boldsymbol{\Sigma}_{tot}^{-1} - (\boldsymbol{\Sigma}_{wc} + 2\boldsymbol{\Sigma}_{ac})^{-1})\boldsymbol{\mu}. \qquad (8) \end{aligned}$$

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

$$\begin{aligned} s &= \mathbf{w}^T\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) \\ &= \begin{bmatrix} \mathrm{vec}(\boldsymbol{\Lambda}) \\ \mathrm{vec}(\boldsymbol{\Gamma}) \\ \mathbf{c} \\ k \end{bmatrix}^T \begin{bmatrix} \mathrm{vec}(\boldsymbol{\phi}_1\boldsymbol{\phi}_2^T + \boldsymbol{\phi}_2\boldsymbol{\phi}_1^T) \\ \mathrm{vec}(\boldsymbol{\phi}_1\boldsymbol{\phi}_1^T + \boldsymbol{\phi}_2\boldsymbol{\phi}_2^T) \\ \boldsymbol{\phi}_1 + \boldsymbol{\phi}_2 \\ 1 \end{bmatrix}. \qquad (9) \end{aligned}$$

Hence, we have obtained a generative generalized linear classifier [8], where the probability for a same-speaker trial can be computed from the log-likelihood ratio score using the sigmoid activation function as

$$p(\mathcal{H}_s|\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \sigma(s) = (1 + \exp(-s))^{-1}. \tag{10}$$

Here, we have assumed equal priors for both hypotheses. To allow for different priors, we can simply adjust the constant $k$ in the vector of weights by adding $\mathrm{logit}(p(\mathcal{H}_s))$.

## 4. DISCRIMINATIVE CLASSIFIERS

In this section, we describe how we train the weights $\mathbf{w}$ directly, in order to discriminate between same-speaker and different-speaker trials, without having to explicitly model the distributions of i-vectors. To represent a trial, we keep the same expansion $\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ as defined in (9). Hence, we reuse the functional form for computing verification scores that provided excellent results with generative PLDA. We consider two standard discriminative linear classifiers, namely logistic regression and SVMs.

### 4.1. Objective functions

The set of training examples, which we continue referring to as training trials, comprises both different-speaker and same-speaker trials. Let us use the coding scheme $t \in \{-1, 1\}$ to represent labels for the different-speaker, and same-speaker trials, respectively. Assigning each trial a log-likelihood ratio $s$ and the correct label $t$, the log probability of recognizing the trial correctly can be expressed as

$$\log p(t|\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = -\log(1 + \exp(-st)). \tag{11}$$

This is easy to see from equation (10) and recalling that $\sigma(-s) = 1 - \sigma(s)$. In the case of logistic regression, the objective function to maximize is the log probability of correctly classifying all training examples, i.e. the sum of expressions (11) evaluated for all training trials. Equivalently, this can be expressed by minimizing the cross-entropy error function, which is a sum over all training trials

$$E(\mathbf{w}) = \sum_{n=1}^{N} \alpha_n E_{LR}(t_n s_n) + \frac{\lambda}{2}\|\mathbf{w}\|^2, \tag{12}$$

where the logistic regression loss function

$$E_{LR}(ts) = \log(1 + \exp(-ts)) \tag{13}$$

is simply the negative log probability (11) of correctly recognizing a trial. We have also added the regularization term $\frac{\lambda}{2}\|\mathbf{w}\|^2$, where $\lambda$ is a constant controlling the tradeoff between the error function and the regularizer. The coefficients $\alpha_n$ allow us to weight individual trials. Specifically, we use them to assign different weights to same-speaker and different-speaker trials. This allows us to select a particular operating point, around which we want to optimize the performance of our system without relying on the proportion of same- and different-speaker trials in the training set. The advantage of using the cross-entropy objective for training is that it reflects performance of the system over a wide range of operating points (around the selected one). For this reason, a similar function was also proposed as a performance measure for the speaker verification task [9]. Another advantage of using the logistic regression classifier is its probabilistic nature: It trains the weights so that the score $s = \mathbf{w}^T \boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ can be interpreted as the log-likelihood ratio between hypotheses $\mathcal{H}_s$ and $\mathcal{H}_d$.

Taking (12) and replacing $E_{LR}(ts)$ with hinge loss function

$$E_{SV}(ts) = \max(0, 1 - ts), \tag{14}$$

we obtain an SVM, which is a classifier traditionally understood to maximize the margin separating class samples. Alternatively, one can see the hinge loss function as a piecewise approximation to the logistic regression loss function. Therefore, one can assume that the score $s = \mathbf{w}^T \boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ obtained from an SVM classifier will still be a reasonable approximation to the log-likelihood ratio (after a linear calibration).

### 4.2. Gradient evaluation

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

$$\nabla E(w) = \sum_{n=1}^{N} \alpha_n \frac{\partial E(t_n s_n)}{\partial s_n} \frac{\partial s_n}{\partial \mathbf{w}} + \lambda \mathbf{w}, \tag{15}$$

where the derivation of the loss function $E(t_n s_n)$, w.r.t. score $s_n$, depends on the particular choice of the loss function. For the logistic regression loss function, it is defined as

$$\frac{\partial E_{LR}(ts)}{\partial s} = -t\sigma(-ts) \tag{16}$$

while for the hinge loss function it becomes

$$\frac{\partial E_{SV}(ts)}{\partial s} = \begin{cases} 0 & \text{if} \quad ts \geq 1 \\ -t & \text{otherwise.} \end{cases} \tag{17}$$

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

$$\frac{\partial s}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}} \mathbf{w}^T \boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = \boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2). \tag{18}$$

### 4.3. Efficient score and gradient evaluation

Given a trained classifier, we can obtain a verification score for a trial by forming the expanded vector $\boldsymbol{\varphi}(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2)$ and computing the dot product (9). However, as we have already seen, the same score can be obtained using the two original i-vectors $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2$ and using the formula (7), which is both memory and computationally efficient. Now, consider two sets of i-vectors stored as columns of the matrices $\boldsymbol{\Phi}_e$ and $\boldsymbol{\Phi}_t$. For illustration, let us call these sets enrollment and test trials, although they play symmetrical roles in our scoring scheme. We can efficiently score each enrollment trial against each test trial and obtain the full matrix of scores as

$$\begin{aligned} \mathbf{S} \;=\; & 2\boldsymbol{\Phi}_e^T \boldsymbol{\Lambda} \boldsymbol{\Phi}_t \\ & +((\boldsymbol{\Phi}_e^T \boldsymbol{\Gamma}) \circ \boldsymbol{\Phi}_e^T)\mathbf{1}\mathbf{1}^T + \mathbf{1}\mathbf{1}^T(\boldsymbol{\Phi}_t \circ (\boldsymbol{\Gamma}\boldsymbol{\Phi}_t)) \\ & +\boldsymbol{\Phi}_e^T \mathbf{c}\mathbf{1}^T + \mathbf{1}\mathbf{c}^T \boldsymbol{\Phi}_t + k\mathbf{1}\mathbf{1}^T, \end{aligned} \tag{19}$$

where $\circ$ denotes the Hadamard, or "entrywise" product. Similarly, the naïve way of evaluating the gradient would be to explicitly expand every training trial and then to apply equations (15) to (18). However, again taking into account the functional form for computing scores (7), the gradient can be evaluated much more efficiently without any need for explicit trial expansion. Let all the i-vectors, which we have available for training, be stored in columns of a matrix $\boldsymbol{\Phi}$. Now consider forming a training trial using every possible pair of i-vectors from the matrix. Let $s_{ij}$ be the score for the trial formed by the $i$-th and $j$-th columns of $\boldsymbol{\Phi}$ calculated using the parameters $\mathbf{w}$ for which we wish to evaluate the gradient. Let $t_{ij}$ and $\alpha_{ij}$ be the corresponding label and trial weight, respectively. Further, let $d_{ij}$ be the corresponding derivation of loss function $E(t_{ij} s_{ij})$ w.r.t. the score $s_{ij}$ given in (16) or (17) depending on the loss function used. The gradient can now be efficiently evaluated as

$$\nabla E(\mathbf{w}) = \begin{bmatrix} \nabla_{\Lambda} L \\ \nabla_{\Gamma} L \\ \nabla_{c} L \\ \nabla_{k} L \end{bmatrix} = \begin{bmatrix} 2 \cdot \operatorname{vec}\left(\boldsymbol{\Phi}\mathbf{G}\boldsymbol{\Phi}^T\right) \\ 2 \cdot \operatorname{vec}\left(\boldsymbol{\Phi}[\boldsymbol{\Phi}^T \circ (\mathbf{G}\mathbf{1}\mathbf{1}^T)]\right) \\ 2 \cdot \mathbf{1}^T[\boldsymbol{\Phi}^T \circ (\mathbf{G}\mathbf{1}\mathbf{1}^T)] \\ \mathbf{1}^T\mathbf{G}\mathbf{1} \end{bmatrix} + \lambda \mathbf{w} \tag{20}$$

where elements of matrix $\mathbf{G}$ are $g_{ij} = d_{ij} \cdot \alpha_{ij}$.

## 5. EXPERIMENTS

The i-vector extractor and the baseline PLDA system is taken from the ABC system submitted to NIST SRE 2010 evaluation [10]. The i-vector extractor uses 60-dimensional cepstral features and a 2048-component full covariance GMM. The UBM and i-vector extractor are trained on NIST SRE 2004, 2005 and 2006, Switchboard and Fisher data. All PLDA systems and discriminative classifiers are trained using 400 dimensional i-vectors extracted from 21663 segments from 1384 female speakers and 16969 segments from 1051 male speakers from NIST SRE 2004, NIST SRE 2005, NIST SRE 2006, Switchboard II Phases 2 and 3, and Switchboard Cellular Parts 1 and 2. Table 1 presents results for the extended condition 5 (tel-tel)

|  | Female Set |  |  | Male Set |  |  | Pooled |  |  |
|---|---|---|---|---|---|---|---|---|---|
| System | minDCF | oldDCF | EER | minDCF | oldDCF | EER | minDCF | oldDCF | EER |
| PLDA | 0.40 | 0.15 | 3.57 | 0.42 | 0.13 | 2.86 | 0.41 | 0.14 | 3.23 |
| LR | 0.40 | 0.12 | 2.94 | 0.39 | 0.10 | 2.22 | 0.40 | 0.11 | 2.62 |
| SVM | 0.39 | 0.11 | 2.35 | 0.31 | 0.08 | 1.55 | 0.37 | 0.10 | 1.94 |
| HT-PLDA | 0.34 | 0.11 | 2.22 | 0.33 | 0.08 | 1.47 | 0.34 | 0.10 | 1.88 |

**Table 1**. Normalized newDCF, oldDCF and EER for the extended condition 5 (tel-tel) from the NIST SRE 2010 evaluation.

from NIST SRE 2010 evaluation. The reported numbers are Equal Error Rate (EER) and normalized minimum Decision Cost Functions for the two operating points as defined by NIST for the SRE 2008 (oldDCF) and SRE 2010 (newDCF) evaluations [11].

The system denoted as PLDA, which serves as our baseline, is based on a generatively trained PLDA model with a 90-dimensional speaker variability subspace [10]. On telephone data, this configuration was found to give the best newDCF, which was the primary performance measure in the NIST SRE 2010 evaluation, which focused on low false alarm rates. As a tradeoff, the system gives somewhat poorer performance at the oldDCF and EER.

The system denoted as LR is the discriminative linear classifier, where parameters were initialized from the baseline system using (8) and retrained to optimize the logistic regression objective function. We have used the conjugate gradient trust region method [12] as implemented in [13] to numerically optimize the parameters. No regularization was used in this case. Significant improvements compared to the baseline can be observed, especially at oldDCF and EER.

Even larger improvements were observed for the SVM-based classifier, where 10%, 30% and 40% relative improvements over the baseline were obtained for newDCF, oldDCF and EER respectively. The improvements over the LR system can probably be attributed mainly to the presence of the regularization term. Often, SVM classifiers are trained using a solver to the dual problem, where a Gram matrix needs to be evaluated. The Gram matrix is a matrix comprising dot products between every pair of training examples, which are the trials in our case. Since we decided to construct a training trial for every pair of i-vectors, the size of the Gram matrix would be unmanageably large (the number of training i-vectors to the 4th power). Therefore, we train a linear SVM by again solving the primal problem using a solver [14], which makes use of the efficient evaluation of gradient. To make SVM regularization effective, we have found that it is necessary to first normalize input i-vectors using within-class covariance normalization (WCCN) [1], i.e. to normalize i-vectors to have identity within-class covariance matrix. More details on the SVM-based system described in this paper can be found in our parallel paper [15].

Finally, for comparison, we also include results with Heavy-tailed PLDA (HT-PLDA) [2], which are so far the best results we have obtained with the same set of training and test i-vectors. In heavy-tailed PLDA, speaker and intersession variability are modeled using Student's $t$, rather than Gaussian distributions. In our system, the dimensionality of i-vectors was first reduced from 400 to 120 and the final vectors were modeled with full-rank speaker and intersession subspaces. Nevertheless, the price paid for the excellent results obtained with heavy-tailed PLDA is the very computationally demanding score evaluation. As we can see, competitive results can be obtained with our discriminatively trained models, for which the score evaluation is several orders of magnitude faster.

## 6. CONCLUSIONS

Recent advances in speaker verification build on i-vector extraction and Probabilistic Linear Discriminant Analysis (PLDA). In this paper, we have proposed to use a PLDA-like functional for evaluating the speaker verification score for a pair of i-vectors representing a trial. However, estimation of the function parameters is based on a discriminative rather than a generative training criterion. We have shown the benefit of using the objective function to directly address the task in speaker verification: discrimination between same-speaker and different-speaker trials. On the NIST SRE 2010 evaluation task, our results show a significant (up to 40%) relative improvement from this approach, compared to a baseline that uses a generatively trained PLDA model.

In future work, we would like to test our method on additional conditions beyond the telephone speech, and to develop techniques for adapting the trained system to be able to cope with new channel conditions. Various methods for regularizing logistic regression training are also worth investigating. We would also like to experiment with models based on more general forms of the PLDA model. Functional forms for verification scores derived from PLDA with low-rank speaker or channel subspaces would allow us to control the number of trainable parameters. Another interesting alternative would be a functional form that would more closely simulate the heavy-tailed PLDA generative model [2], which is currently providing better performance than PLDA based on Gaussian distributions.

## 7. REFERENCES

[1] N. Dehak, P. Kenny, et al., "Front–end factor analysis for speaker verification," in *IEEE Trans. on Audio, Speech and Lang. Process.*, 2010.

[2] P. Kenny, "Bayesian speaker verification with heavy–tailed priors," keynote presentation, Proc. of Odyssey 2010, June 2010.

[3] W.M. Campbell, D.E. Sturim, D.A. Reynolds, and A. Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation," May 2006, vol. 1, pp. I –I.

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

[5] L. Burget et al., "Robust speaker recognition over varying channels," in *Johns Hopkins University CLSP Summer Workshop Report*, 2008, www.clsp.jhu.edu/workshops/ws08/documents/jhu_report_main.pdf.

[6] P. Kenny et al., "Joint factor analysis versus eigenchannes in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

[7] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *11th International Conference on Computer Vision*, 2007, pp. 1–8.

[8] C. M. Bishop, *Pattern Recognition and Machine Learning*, chapter 4.2, Springer, 2006.

[9] N. Brümmer and J. A. du Preez, "Application-independent evaluation of speaker detection," *Computer Speech & Language*, vol. 20, no. 2-3, pp. 230–275, 2006.

[10] N. Brummer, L. Burget, P. Kenny, et al., "ABC system description for NIST SRE 2010," in *Proc. NIST 2010 Speaker Recognition Evaluation*.

[11] NIST, "The NIST year 2008 and 2010 speaker recognition evaluation plans," http://www.itl.nist.gov/iad/mig/tests/sre.

[12] Jorge Nocedal and Stephen J. Wright, *Numerical Optimization*, Springer, August 2000.

[13] E. de Villiers and N. Brümmer, "BOSARIS toolkit," https://sites.google.com/site/bosaristoolkit/.

[14] C.H. Teo, A. Smola, et al., "A scalable modular convex solver for regularized risk minimization," in *Proc. of KDD*, 2007, pp. 727–736.

[15] S. Cumani, N. Brummer L. Burget, , and P. Laface, "Fast discriminative speaker verification in the i-vector space," submitted to Proc. of ICASSP 2011, Prague.

# iVector Fusion of Prosodic and Cepstral Features for Speaker Verification

Marcel Kockmann[1] and Luciana Ferrer[2] and Lukáš Burget[1] and Jan "Honza" Černocký[1]

[1]Brno University of Technology, Speech@FIT, Czech Republic
[2]Speech Technology and Research Laboratory, SRI International, Menlo Park, CA, USA

## Abstract

In this paper we apply the promising iVector extraction technique followed by PLDA modeling to simple prosodic contour features. With this procedure we achieve results comparable to a system that models much more complex prosodic features using our recently proposed SMM-based iVector modeling technique. We then propose a combination of both prosodic iVectors by joint PLDA modeling that leads to significant improvements over individual systems with an EER of 5.4% on NIST SRE 2008 telephone data. Finally, we can combine these two prosodic iVector front ends with a baseline cepstral iVector system to achieve up to 21% relative reduction in new DCF.

**Index Terms**: speaker verification, prosody, JFA, iVector, SMM, fusion

## 1. Introduction

High-level information has been used for over a decade to further enhance short-time, cepstral-based speaker verification systems. Many approaches make use of acoustic attributes of speech prosody that mainly involve variations in syllable length, loudness, and pitch. In recent NIST Speaker Recognition Evaluations [1, 2], two families of prosodic feature sets were presented. One family corresponds to syllable-based, non-uniform extraction region features (SNERFs) [3], which are highly complex prosodic features originally proposed by SRI. These features in combination with specialized parameterization methods and support vector machine (SVM) modeling [4] result in a very good prosodic system.

Another family of systems uses a set of very simple prosodic features, originally proposed for language identification [5]. These features model the temporal trajectory of pitch and energy over the time span of a syllable. Joint Factor Analysis (JFA) modeling for these features was originally proposed by [6] and showed very promising results. This framework for prosodic modeling has been adopted by several sites and investigated thoroughly [7, 8]. The main reason for its success lies in JFA modeling, which is capable of coping with the problem of speaker and session variability in Gaussian mixture model (GMM)-based speaker verification [9] and has become the de facto standard for modeling low- and high-level features.

Moreover, excellent results on cepstral features were obtained with a simplified variant of JFA [10], where separate subspaces for channel and speaker variability are replaced by a single subspace covering the total variability. This model can be used to extract compact low-dimensional feature vectors representing a whole utterance, often called iVectors. Based on this idea, we proposed a framework where the subspace modeling technique normally used to model means of GMMs is adapted to model occupation counts using a multinomial model. This so-called Subspace Multinomial Model (SMM) [11] is applicable to the complex SNERFs to extract iVectors.

Probabilistic Linear Discriminant Analysis (PLDA) [12] has been proposed to model the speaker and channel variability in both types of iVectors, directly generating likelihood ratios for the trials [13, 14]. iVector modeling of SNERFs by SMMs with successive PLDA has been shown to give the best results for a prosodic speaker verification system so far [15].

To date, the iVector approach – using a total variability subspace followed by PLDA – has not been used (to our knowledge) for the simple prosodic features that are usually modeled by JFA.

In this paper, we present results on the prosodic JFA system as presented by Brno University of Technology in SRE 2010 and apply iVector modeling and PLDA back end to the same features. We show that the iVector approach is superior to the standard JFA modeling even for simple prosodic features.

In this way we have two diverse prosodic systems that achieve similar performance on our test sets: an iVector system that models means of GMMs based on simple well-defined prosodic features and an iVector system that models counts of multinomial distributions based on SNERFs. A combination of both systems seems relevant due to their complementary nature in terms of features and modeling. We propose an elegant way of combining these systems by simple concatenation of individual iVectors followed by a single joint PLDA model. This combination achieves an equal error rate (EER) of 5.4% on our NIST SRE 2008 telephone test set, a 23% gain over the best of the two systems.

Justification for use of a higher-level systems usually lies in an overall improvement by fusion with a cepstral baseline system. Usually, combination of low- and high-level systems is done by score-level fusion using a separate development set to train the fusion parameters. As the best-performing cepstral systems to date are also based on iVector modeling followed by PLDA modeling [13, 14, 16], we are inspired by the successful combination of two prosodic iVector front ends to further combine the cepstral and prosodic systems in the same manner. We achieve a relative reduction in terms of the challenging new detection cost function (DCF) [2] of 17% for SRE 2010 data and 21% for SRE 2008 data. The iVector combination consistently outperforms standard score-level fusion (11% and 13%) with no need for a separate development set to train the fusion parameters.

This work was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory (ARL). All statements of fact, opinion, or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI, or the U. S. Government. The work was also partly supported by Czech Ministry of Education project No. MSM0021630528, and Grant Agency of Czech Republic project No. GP102/09/P635. Marcel Kockmann was supported by SVOX Deutschland GmbH.

## 2. Prosodic features

This section describes the two prosodic feature sets used in the paper.

### 2.1. DCT contour features

The DCT contour feature generation closely follows the description in [7]. The features incorporate duration, pitch and energy measurements. Pitch and energy values are estimated every 10 ms, and energy is further normalized by its maximum. The temporal trajectory of pitch and energy is modeled by a discrete cosine transform (DCT), over a fixed frame long temporal window of 300 ms, with a 50 ms frame shift. The first six DCT coefficients of both pitch and energy trajectories form a fixed-length feature vector. Only voiced frames (where pitch is detected) are used to estimate the DCT. Duration information measured as the number of voiced frames within the 30-frame interval is appended and treated as a continuous value when modeling the distributions.

### 2.2. SNERF features

We use SNERFs, which are syllable-based prosodic features based on estimated pitch, energy, and duration information. Characteristics like minimum, maximum, mean, and slope of pitch and energy trajectories are extracted for each detected syllable in an utterance and its nucleus, as well as duration of onset, nucleus, and coda of the syllable. All values are further normalized with different techniques and form several hundred features for each syllable. 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 [17]) on the phone-level alignments. Detailed information on SNERFs is given in [3].

We use 182 basic features that are extracted for each syllable. Furthermore, temporal dependencies are modeled by constructing small vectors concatenating features from consecutive syllables and pauses. These so-called tokens are formed for each basic feature by concatenating as many as three values (feature values and duration of pauses; more details are given in [4]). Nine different n-gram tokens are used.

The SNERFs are parameterized by use of GMMs. This can be seen as a soft binning of each SNERF value into a meaningful set of discrete classes and makes it possible to accumulate soft counts for all SNERFs and tokens extracted for one utterance (for details see [4]).

## 3. Subspace models for prosodic features

The basic assumption in subspace modeling is that the natural parameters of a model usually live in a much smaller subspace than the full parameter space. This subspace can be learned by introducing latent variables in the model.

### 3.1. iVectors based on GMMs

The classical formulation of JFA for speaker verification [9] assumes that the concatenated mean vectors $\phi_{\text{GaussJFA}}$ of a GMM are distributed according to a subspace model with separate subspaces for speaker and channel variability:

$$\phi_{\text{GaussJFA}} = \mathbf{m} + \mathbf{V}\mathbf{y} + \mathbf{U}\mathbf{x}, \tag{1}$$

where $\mathbf{m}$ is a speaker- and channel-independent supervector, and $\mathbf{V}$ and $\mathbf{U}$ span linear subspaces (for speaker and channel variability) in the original mean parameter space. The components of $\mathbf{y}$ and $\mathbf{x}$ are the low-dimensional latent variables corresponding to the speaker and channel subspaces.

A simplified variant of JFA [10] assumes that speaker and channel subspaces are not decoupled and uses only one subspace covering the total variability in an utterance:

$$\phi_{\text{GaussIV}} = \mathbf{m} + \mathbf{T}\mathbf{w}. \tag{2}$$

Again, $\mathbf{T}$ spans a linear subspace in the original mean parameter space and the components of $\mathbf{w}$ are the low-dimensional latent variables corresponding to the total variability subspace. The low-dimensional vectors $\mathbf{w}$ are also known as iVectors.

In the latter approach, the JFA-like model serves only as the extractor of the vectors $\mathbf{w}$, which can be seen as low-dimensional fixed-size representations of utterances, and which are in turn used as inputs to another classifier.

Both techniques, the JFA (*GaussJFA*) as well as the iVector modeling (*GaussIV*), are applicable to mean supervectors of GMMs trained on the low-dimensional well-defined DCT features as presented in Section 2.1. All model parameters are trained using an expectation-maximization (EM) algorithm [9].

### 3.2. iVectors based on multinomial distributions

The weights of a GMM can also be modeled under the subspace paradigm. To do this, we consider the individual mixture components in the GMM to be discrete classes which can be modeled using a multinomial distribution. Similar to *GaussIV*, SMM assumes that there is a low-dimensional subspace of the parameter space in which the parameters of the multinomial distributions for individual utterances live. The probability $\phi_{\text{MultinIV}}$ of $c$th class of the multinomial distribution is given by

$$\phi_{\text{MultinIV}} = \frac{\exp(m + \mathbf{t}_c\mathbf{w})}{\sum_{i=1}^{C} \exp(m + \mathbf{t}_i\mathbf{w})}, \tag{3}$$

where $\mathbf{w}$ is a latent variable and $\mathbf{t}_c$ is the $c$th row of subspace matrix $\mathbf{T}$, which spans a linear subspace in the log-probability domain. Due to the softmax function, this corresponds to a possibly nonlinear subspace in the simplex that the multinomial distributions live in.

Given the parameters $\mathbf{m}$ and $\mathbf{T}$ we can extract $\mathbf{w}$ vectors (which we will also call iVectors) for new data. Similar to the *GaussIV* system, the SMM is used as a feature extractor and each iVector can be seen as a low-dimensional representation of the whole utterance.

This technique (*MultinIV*) can be used to model soft counts of high-dimensional, heterogeneous SNERFs as presented in Section 2.2. See [11] for further details of how all SNERFs can be represented using a single low-dimensional iVector and how the model parameters are trained using an iterative optimization scheme.

### 3.3. PLDA modeling of iVectors

The fixed-length iVectors extracted per utterance (from the *GaussIV* as well as from the *MultinIV* model) can now be used as input to a pattern recognition algorithm. Note that unlike in 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 effect, and intersession compensation must be eventually considered during classification.

For speaker verification a PLDA model [12] has been proposed to provide a probabilistic framework for modeling speaker and intersession variability in the iVector space. Model parameters can be trained using an EM algorithm [13]. Using the PLDA model, one can directly evaluate the log-likelihood ratio for the hypothesis test corresponding to "the two iVectors were generated by the same speaker or not". This can be evaluated analytically, and scoring can be performed very efficiently as described in [14].

# 4. Experiments

This section describes the experimental setup and results for the individual prosodic systems and for the combination of these systems with each other and with a baseline cepstral system.

## 4.1. Data

Results are presented on the telephone core conditions of the NIST Speaker Recognition Evaluations 2008 [1] (*dev*) and 2010 [2] (*eval*). Trials involve English conversational speech recorded over various telephone channels. Our development set is based on the original NIST SRE 2008 evaluation set, but was extended to include about two orders of magnitude more impostor samples, to adjust for the new DCF point. It includes 1,154 target and 1,516,837 nontarget trials. Our evaluation set corresponds to the official extended condition 5 of NIST SRE 2010 and contains 7,169 target and 408,950 nontarget trials.

Training of background, subspace, and PLDA models is performed on data from Switchboard corpora as well as NIST SRE 2004 – 2006 corpora. This set includes 13,482 recordings from 752 male and 16,782 recordings from 963 female speakers.

## 4.2. Prosodic systems

Experiments are carried out to evaluate the performance of the iVector modeling approach for the simple DCT features. For both, the *GaussJFA* and the *GaussIV* systems, we extract 13-dimensional DCT contour features (1 duration, 6 pitch and 6 energy values) and train gender-dependent multivariate universal background models (UBMs) with 512 Gaussian components and diagonal covariances. The *GaussJFA* and the *GaussIV* models are trained using sufficient statistics extracted for all background data using the same UBMs. For the *GaussJFA* model we train 100-dimensional speaker subspace $\mathbf{V}$ and 50-dimensional channel subspace $\mathbf{U}$. For the *GaussIV* model we train 300-dimensional total variability subspace $\mathbf{T}$ on the same data. These subspace sizes were found optimal in earlier experiments. The *GaussJFA* model is evaluated directly by log-likelihood ratio using a fast scoring technique [18] followed by zt-norm. The extracted DCT iVectors for all background data are used to train a full rank PLDA model. The PLDA model is then used to evaluate the log-likelihood ratio for speaker trials. Figure 1 shows results for the two DCT-based systems (green markers). The *DCT-GaussIV* system with PLDA (square) clearly outperforms the *DCT-GaussJFA* system (triangle) on all operating points on both test sets.

To compare the simple *DCT-GaussIV* system with the best prosodic system presented so far [15], we train a *SNERF-MultinIV* system on the same setup. The SMM models an ensemble of 1,638 multinomial distributions representing 9 different n-gram tokens of 182 individual SNERFs. We obtain 300 dimensional iVectors. While the *SNERF-MultinIV* system (blue diamonds in Figure 1) is still superior on both test sets for EER and old DCF, we achieve better results with the *DCT-GaussIV* system on both test sets in terms of new DCF.

As both prosodic systems perform very well, but are significantly different in terms of features as well as 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 feature vector per utterance (what we call iVector), it is possible to simply concatenate the iVectors resulting from these diverse models and to model them jointly with a PLDA model. We train a single full-rank PLDA model on 600-dimensional iVectors. The effectivity of the joint modeling of complementary iVectors can be observed in Figure 1. The combination of *DCT-GaussIV* and *SNERF-MultinIV* iVectors (cyan hexagons) results in significant improvement over the best individual system on all operating points on both test sets, achieving an EER of 5.4% and a new DCF of 0.72 on 2008 data, which are (to our knowledge) the best results reported for a purely prosodic system.

## 4.3. Combination with cepstral baseline system

Our baseline system is a cepstral iVector system followed by a PLDA model (*CEP-GaussIV*). This system was the best-performing individual system from the ABC NIST SRE 2010 submission [16]. It is based on 60-dimensional cepstral features and a 2048-component full covariance UBM. Four hundred-dimensional iVectors are used and the dimension is further reduced to 200 by standard LDA and normalized by their length[1] before PLDA modeling. The first row of Table 1 gives the results for our two data sets[2].

Again, the iVector nature of our 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. First, we apply an LDA reduction to 200 dimensions and length normalization to both 300-dimensional sets of prosodic iVectors. In this way we have three same sized sets of 200 dimensional iVectors (one cepstral and two prosodic). Next, we concatenate the cepstral iVectors separately with each of our prosodic iVectors to obtain two sets of four hundred-dimensional iVectors. Then we train a standard PLDA model with full rank of 400 for each type of combination. The second and third row of Table 1 give the results for these combinations. We see that we can achieve significant improvements for both *iVector fusions* of cepstral and prosodic features. Finally, we concatenate all three iVector types (one cepstral and two prosodic) and train a PLDA model with full rank of 600. The fourth row of Table 1 gives the results for this combination. We achieve further improvements leading to reductions as high as 21% relative on the challenging new DCF measure.

As a last experiment we compare this approach to the conventional score-level fusion. For this purpose we train a linear logistic regression [19] to fuse the three individual system scores on the development set and apply this fusion to the evaluation set. The last row of Table 1 indicates that consistent gains are also achieved by score-level fusion (as high as 13% on new DCF), but joint PLDA training of concatenated iVectors remains superior. iVector fusion of the cepstral system and the simple prosodic *DCT-GaussIV* system already outperforms the score-level fusion of all three systems.

# 5. Conclusions and Lookout

We present the first results on the use of total variability modeling of the mean supervector space for a set of prosodic features. We show that this iVector approach outperforms the standard JFA approach originally proposed for these features. We note that this improvement over JFA is observed only when the iVectors are modeled using the PLDA back end. No gain was observed during SRE 2010 system development [16] when iVectors were modeled with simpler scoring techniques [6].

[1] This pre-processing of iVectors is very helpful for cepstral iVectors but did not show any improvement for our prosodic iVectors

[2] We are aware that better results are reported in the literature, simply by training the PLDA on more data, which we did not have for SNERFs.

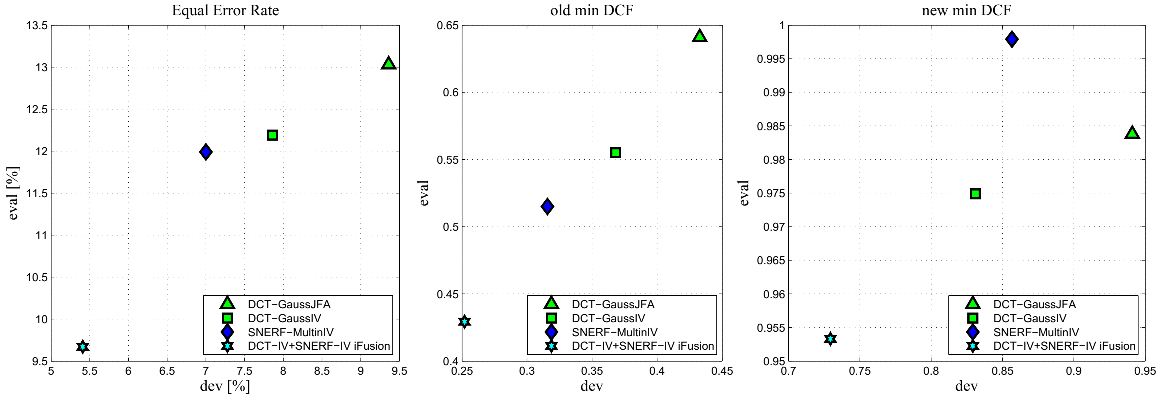


*Figure 1:* Results for SRE 2008 (dev) versus SRE 2010 (eval) in terms of EER, old DCF and new DCF, from left to right, for three different prosodic systems and combination of the two best.

| System | DEV SRE 2008 | | | EVAL SRE 2010 | | |
|---|---|---|---|---|---|---|
| | EER | old DCF | new DCF | EER | old DCF | new DCF |
| Cepstral iVector system *CEP-GaussIV* | 2.02 | 0.090 | 0.471 | 3.14 | 0.155 | 0.504 |
| Concatenated *CEP-GaussIV + DCT-GaussIV* | 1.69 | 0.080 | 0.400 | 2.72 | 0.136 | 0.431 |
| Concatenated *CEP-GaussIV + SNERF-MultinIV* | **1.65** | 0.080 | 0.389 | 2.74 | 0.134 | 0.444 |
| Concatenated *CEP-GaussIV + DCT-GaussIV + SNERF-MultinIV* | 1.70 | **0.075** | **0.368** | **2.63** | **0.129** | **0.421** |
| Score fusion *CEP-GaussIV + DCT-GaussIV + SNERF-MultinIV* | 1.92 | 0.078 | 0.406 | 3.09 | 0.149 | 0.447 |

*Table 1: Results for single cepstral baseline system (CEP-GaussIV) and for combinations with one or two prosodic iVector systems.*

Furthermore, we present combination results of two prosodic systems, one where iVectors based on GMMs are used to model simple DCT features extracted from uniform regions and another one 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 modeling levels. We show gains on the order of 20% when combining these two systems with respect to the single best. The combination is performed using an iVector-level fusion: the individual iVectors for the two systems are concatenated and the joint iVector is modeled using PLDA. 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.

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

## 6. References

[1] NIST, “The NIST year 2008 speaker recognition evaluation plan,” 2008. [Online]: http://www.itl.nist.gov/iad/mig//tests/sre/2008

[2] ——, “The NIST year 2010 speaker recognition evaluation plan,” 2010. [Online]: http://www.itl.nist.gov/iad/mig//tests/sre/2010

[3] E. Shriberg *et al.*, “Modeling prosodic feature sequences for speaker recognition,” *Speech Communication*, Jan 2005.

[4] L. Ferrer *et al.*, “Parameterization of prosodic feature distributions for SVM modeling in speaker recognition,” *Proc. ICASSP, Taipei*, vol. 4, pp. 233–236, 2007.

[5] C.-Y. Lin *et al.*, “Language identification using pitch contour information,” *Proc. ICASSP 2005, Philadelphia, PA*, pp. 601–604, 2005.

[6] N. Dehak *et al.*, “Modeling prosodic features with joint factor analysis for speaker verification,” *Audio*, Jan 2007.

[7] M. Kockmann *et al.*, “Investigations into prosodic syllable contour features for speaker recognition,” *Proc. of ICASSP, Dallas*, pp. 1–4, Sep 2010.

[8] L. Ferrer *et al.*, “A comparison of approaches for modeling prosodic features in speaker recognition,” *Proc. ICASSP, Dallas*, 2010.

[9] P. Kenny *et al.*, “A study of inter-speaker variability in speaker verification,” *IEEE Trans. Audio*, Jan 2008.

[10] N. Dehak *et al.*, “Front-end factor analysis for speaker verification,” *IEEE Transactions on Audio, Speech and Language processing*, pp. 1–23, Jul 2009.

[11] M. Kockmann *et al.*, “Prosodic speaker verification using subspace multinomial models with intersession compensation,” in *Proc. Interspeech, Tokyo*, 2010.

[12] S. J. D. Prince, “Probabilistic linear discriminant analysis for inferences about identity,” in *ICCV*, 2007.

[13] P. Kenny, “Bayesian speaker verification with heavy tailed priors,” in *Keynote presentation, Odyssey*, 2010.

[14] L. Burget *et al.*, “Discriminatively trained probabilistic linear discriminant analysis for speaker verification,” in *ICASSP*, 2011.

[15] M. Kockmann *et al.*, “Recent progress in prosodic speaker verification,” in *Proc. ICASSP, Prague*, 2011.

[16] N. Brummer *et al.*, “ABC system description for NIST SRE 2010,” in *Proc. NIST 2010 Speaker Recognition Evaluation*. Brno University of Technology, 2010, pp. 1–20.

[17] L. Ferrer, “Statistical modeling of heterogeneous features for speech processing tasks,” Ph.D. dissertation, Stanford University, 2009.

[18] O. Glembek *et al.*, “Comparison of scoring methods used in speaker recognition with joint factor analysis,” *Proc. of ICASSP, Taipei*, 2009.

[19] E. de Villiers *et al.*, “BOSARIS toolkit,” 2010. [Online]: http://sites.google.com/site/bosaristoolkit

# RECENT PROGRESS IN PROSODIC SPEAKER VERIFICATION

Marcel Kockmann[1], Luciana Ferrer[2], Lukáš Burget[1], Elizabeth Shriberg[2] and Jan “Honza” Černocký[1]

[1]Brno University of Technology, Speech@FIT, Czech Republic
[2]Speech Technology and Research Laboratory, SRI International, Menlo Park, CA, USA

## ABSTRACT

We describe recent progress in the field of prosodic modeling for speaker verification. In a previous paper, we proposed a technique for modeling syllable-based prosodic features that uses a multinomial subspace model for feature extraction and within-class covariance normalization or linear discriminant analysis for session variability compensation. In this paper, we show that performance can be significantly improved with the use of probabilistic linear discriminant analysis (PLDA) for session variability compensation. This system does not require score normalization. We report an equal error rate below 7% on a NIST 2008 task. To our knowledge, this is the best reported result to date for a prosodic system for speaker recognition. Fusion of this system with a state-of-the-art acoustic baseline system yields 10% relative improvement in the new detection cost function (DCF) as defined by NIST.

***Index Terms***— Prosodic speaker verification, SNERFs, MSM, iVector, PLDA

## 1 INTRODUCTION

Using high-level information to further enhance short-time, cepstral-based speaker verification systems has been popular for several years. In [1], several high-level features (phonetic, prosodic, linguistic, etc.) were leveraged to enhance the Equal Error Rate (EER) on the NIST 2001 speaker recognition evaluation task up to 70% relative. This gain from using high-level features was enabled by the introduction of evaluation conditions with large train and test durations (of 2.5 minutes for testing and up to 8 times that amount for training). High-level features are sparser than lower-level acoustic features and, hence, benefit more from large amounts of data. During subsequent NIST evaluations, challenging new corpora and rapid performance improvements for systems using standard cepstral features generally made gaining an advantage from the fusion of high-level features difficult [2].

Nevertheless, in 2004, high-level features were shown to provide performance gains greater than 30% when combined with a baseline acoustic system on the NIST 2004 tasks [3]. The success was mainly due to SRI’s newly proposed, syllable-based, non-uniform extraction region features (SNERFs) [4]. These features in combination with specialized parameterization methods and Support Vector Machine (SVM) modeling [5] resulted in the best-performing prosodic system at the time. But, the SNERF system was complex and for this reason, was not broadly adopted by the community.

The introduction of joint factor analysis (JFA) [6] for speaker verification brought the performance of acoustic systems for speaker recognition to a new level, leading to improvements on the order of 50% over previous state-of-the-art systems. As a consequence of these dramatic improvements in the baseline performance of speaker recognition systems, obtaining gains from high-level features, particularly if they could not capitalize on the JFA improvements obtained for acoustic systems, was increasingly difficult. A first step in using JFA for prosodic systems was proposed by [7] for a set of very simple prosodic features. This framework for prosodic modeling has been adopted by several sites and investigated thoroughly [8, 9].

Unfortunately, the JFA framework cannot be directly applied to the SNERFs due to their high dimensionality and to the existence of undefined values. In [9], we showed that the SNERF system still outperforms a simpler set of features modeled with JFA. This was our motivation for trying to transfer the underlying idea of JFA – to model speaker and intersession variability in low-dimensional subspaces – to a model that can handle SNERFs. Recently, we presented a theoretic framework for the modeling of SNERFs using a multinomial subspace model (MSM), which achieved very promising results [10].

This paper describes our latest progress in using Probabilistic Linear Discriminant Analysis (PLDA) modeling for session variability compensation of features obtained with MSM. Significant gains are achieved over previous performance, resulting in an equal error rate (EER) of 6.9% on the telephone data of the NIST 2008 Speaker Recognition Evaluation [11]. To our knowledge, these are the best results in the literature for a prosodic speaker verification system. Furthermore, no score normalization techniques are needed. In addition, we present fusion experiments with a state-of-the-art acoustic JFA system showing gains of up to 10% in detection cost function (DCF). A major goal of this paper is to clearly describe the complex system-building process. All important steps – from raw SNERF features to final PLDA modeling – are explained in Section 2. In Section 3, our experimental setup is described and different prosodic systems are evaluated and compared. Fusion results with a baseline acoustic system are also shown. We present our conclusions in Section 4.

## 2 SYSTEM

This section describes the five major steps of the system-building process. All steps are explained using a simplified example. Please refer to the citations for algorithmic descriptions.

### 2.1 Syllable-based NERFs (SNERFs)

We use SNERFs [4], which are syllable-based, non-uniform extraction region features based on F0, energy, and duration information. Characteristics like minimum, maximum, mean, and slope of the

This work was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory (ARL). All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI, or the U. S. Government. The work was also partly supported by European project MOBIO (FP7-214324), Grant Agency of Czech Republic project No. 102/08/0707, Czech Ministry of Education project No. MSM0021630528 and by BUT FIT grant No. FIT-10-S-2. Marcel Kockmann was supported by SVOX Deutschland GmbH.

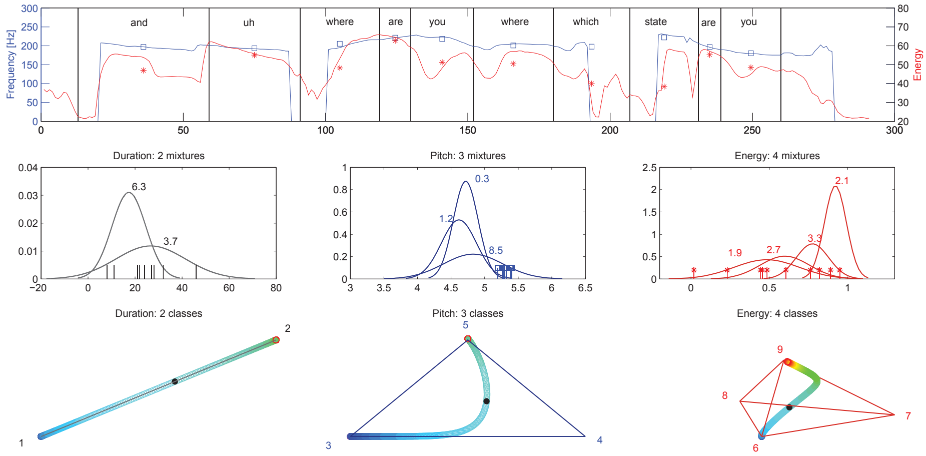


**Fig. 1**. **Top row:** Extraction of three SNERF parameters from a speech segment containing 10 single-syllable words: Syllable duration (determined by black vertical lines), mean pitch value per syllable (blue squares), and mean energy per syllable (red stars). **Middle row:** Parameterization of 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.

pitch and energy trajectories are extracted for each detected syllable in an utterance and for its nucleus, as well as the duration of onset, nucleus and coda of the syllable. All values are further normalized with different techniques, resulting in a few hundred features for each syllable (174 in our current implementation). The syllable segmentation is generated from the output of a large vocabulary continuous speech recognition (LVCSR) system. The phone alignments of the recognized words are used to generate English syllables. Detailed information on SNERFs is given in [4].

Temporal 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 each sequence length, each feature, and each pattern of pause/non-pause, we obtain a separate feature vector. For example, for trigrams, we obtain five different vectors: $(S, S, S)$, $(P, S, S)$, $(S, P, S)$, $(S, S, P)$, $(P, S, P)$ for each feature. Each pair {feature, pattern} determines what we call a *token* (see [5] for details). Our current implementation uses sequences of lengths 1, 2, and 3. The first line of plots in Figure 1 shows an example of the feature extraction process. The segments are given by the syllables found from the ASR output. The pitch (blue curve) and energy (red curve) signals are estimated from the waveform. For our example, we assume that we extract only three features per segment: its duration (from one vertical black line to the next), the mean pitch value (blue squares), and the mean energy value (red stars).

### 2.2 Background GMMs

For each token, we train a separate Gaussian Mixture Model (GMM) 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 an additional parameter for the probability of a feature being undefined. In the first pass, all GMMs are trained using frames with defined features only, where the additional parameter is set to one and the model falls back to a standard GMM. The GMMs are then retrained with all feature vectors, allowing the new parameter to adapt to the data. Details of the modified expectation-maximization algorithm are given in [12]. The second line of Figure 1 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.

### 2.3 Parameterization of SNERF sequences

After training the background models for each token, we gather Gaussian component occupation counts for each utterance (zero order sufficient statistics from the modified EM algorithm [12]). These are accumulated soft counts describing the responsibilities of each individual mixture component toward generating the frames in the utterance. Using these parameters, we transform the sequence of SNERFs (one feature vector per syllable) to fixed length vectors (one vector of statistics per utterance). The values from the exemplified feature extraction process (syllable duration, mean pitch, and mean energy) are further depicted as bars in the middle row of Figure 1. 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.

### 2.4 Multinomial Subspace Model

As a generative model, a multinomial distribution appears as a natural choice for modeling the counts resulting from the previous 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 $\phi_{ec}$ 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 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. The bottom row of Figure 1 illustrates this for our toy example where 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.

We use a Multinomial Subspace Model (MSM) [10] where we assume that the multinomial distributions differ from utterance to utterance. In the case of SNERFs, we need to estimate parameters of many multinomial distributions. Therefore, we search for a way to estimate all the parameters robustly given a limited amount of data available for each utterance. With MSM, we assume that there is a low-dimensional subspace of the parameter space in which the parameters for individual utterances live. For this reason we introduce an explicit latent variable $\mathbf{w}$ through which the probability $\phi_{ec}$ of $c$th class of each multinomial distribution $e$ in the ensemble is given by

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

with $\mathbf{t}_{ec}$ being the $c$th row of $e$th block of subspace matrix $\mathbf{T}$ (size $\sum_{e=1}^{E} C_e \times r$) which spans a linear subspace that might be non-linear in the original parameter space due to the softmax function. Figure 1 shows how the subspace restricts the movement in the full parameter-space in a non-linear way (colored lines). By drawing values for a one-dimensional variable $\mathbf{w}$ from minus infinity to infinity we move in all three simplexes simultanuously along the non-linear, low-dimensional manifolds. Now, all the multinomial distributions corresponding to one utterance can be represented by a low dimensional vector $\mathbf{w}$. This way, we can (1) reduce the number of free parameters to efficiently model differences between individual utterances, and (2) learn dependencies between the individual SNERFs.

The MSM 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 $\mathbf{T}$ we can extract $\mathbf{w}$ vectors (which we will call iVectors) for new data. This way, the MSM is used as a feature extractor and each iVector can be seen as a low-dimensional representation of the whole utterance.

### 2.5 PLDA modeling

For verification of speaker trials we use a special case of Probabilistic Linear Discriminant Analysis (PLDA) [13], a two-covariance model, providing a probabilistic framework where speaker and inter-session variability in the iVectors is modeled using across-class and within-class covariance matrices $\mathbf{\Sigma}_{ac}$ and $\mathbf{\Sigma}_{wc}$. We assume that latent vectors $\mathbf{y}$ representing speakers are distributed according to

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mu, \mathbf{\Sigma_{ac}}) \tag{2}$$

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

$$p(\mathbf{w}|\mathbf{y}) = \mathcal{N}(\mathbf{w}; \mathbf{y}, \mathbf{\Sigma_{wc}}). \tag{3}$$

Model parameters $\mu$, $\mathbf{\Sigma}_{ac}$ and $\mathbf{\Sigma}_{wc}$ are trained using an EM algorithm [14]. Using the PLDA model, one can directly evaluate the log-likelihood ratio for the hypothesis test corresponding to "the two iVectors were generated by the same speaker or not":

$$s = \log \frac{\int p(\mathbf{w}_1|\mathbf{y})p(\mathbf{w}_2|\mathbf{y})p(\mathbf{y})d\mathbf{y}}{p(\mathbf{w}_1)p(\mathbf{w}_2)} \tag{4}$$

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 from different speakers. The integrals can be evaluated analytically and scoring can be performed very efficiently as described in [15].

## 3 EXPERIMENTS AND RESULTS

This section describes our results for three individual prosodic systems, including two previously-proposed systems. We also show results when fusing the prosodic systems with a state-of-the-art cepstral system.

### 3.1 Data

The task used to present results uses data from the NIST 2008 speaker recognition evaluation. The original NIST tasks are extended to include two orders of magnitude more impostor samples. This was done to support the new DCF metric introduced by NIST for the 2010 evaluation [16]. In this paper, we show results only for the telephone condition, in which both training and test samples are given by telephone conversations recorded over a telephone channel. The number of target and impostor samples for this task are 1,108 and 1,453,237, respectively. As background data to train UBMs, JFA, MSM and PLDA we use data from the 2004 and 2005 SRE, 2008 interview development data and from the Switchboard-II corpus.

### 3.2 Prosodic systems

We evaluate three different prosodic systems: (1) a system based on JFA-modeling of means of low-dimensional polynomial features (Prospol) describing pitch and energy trajectories, originally proposed in [7] and further extended and improved as described in [9]; (2) the baseline SNERF system with SVM modeling (SNERF-SVM) of the counts as originally described in [5]; and (3) the recently introduced subspace model [10] with additional PLDA modeling applied to the SNERF counts (SNERF-IV-PLDA).

The Prospol system models a small set of 13 features, including polynomial approximations of the pitch and energy profiles and the duration of the region for three different region definitions: (1) energy valleys (as originally proposed in [7]); (2) uniform windows of 300 msec shifted by 10 msec (as proposed in [8]); and (3) syllable regions (identical to those used for the SNERFs). Further, sequences of length 2 are also modeled. For each region and each sequence length, a separate system is created. The resulting scores are combined with fixed weights determined empirically from development data. The baseline SNERF system directly uses the occupation counts (divided by the number of frames) as features for an SVM model (steps 2.1–2.3). Session variability compensation can be applied to this model using nuisance attribute projection [17], but we found no significant gains from this approach. For the SNERF-IV-PLDA system the occupation counts are used to train an MSM with a subspace dimension $r$=200 following [10]. Next, iVectors are extracted using this model for all background, training and test utterances. The PLDA model is then trained[1] on iVectors extracted for all background data and is used to perform verification between speaker trials. Figure 2 shows the DET curves for the three prosodic systems. Both SNERF systems outperform the Prospol system at all operating points of interest. Further, the proposed modeling technique for the SNERFs is significantly better than the older method based on SVMs for most operating points resulting in an EER of 6.9%. Moreover, the PLDA modeling significantly outperforms the cosine distance scoring with LDA as used in our previous work with MSMs (9% EER) [10].

### 3.3 Acoustic system

The cepstral GMM baseline system uses a 300-3300 Hz bandwidth front-end consisting of 24 Mel filters to compute 20 cepstral coeffi-

[1] We thank Niko Brümmer for providing his PLDA implementation.

**Table 1**. *Relative improvement over cepstral JFA baseline [%].*

| | System | new DCF | old DCF | EER |
|---|---|---|---|---|
| Fusion | Baseline+Prospol | 6.25 | -1.37 | -5.26 |
| | Baseline+SNERF-SVM | 7.21 | 3.70 | 10.53 |
| | Baseline+SNERF-IV-PLDA | 9.62 | 5.08 | 5.27 |

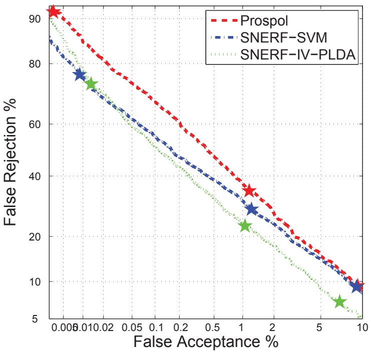


**Fig. 2**. DET curves for the three prosodic systems. The three markers in each line correspond to the new DCF, the old DCF, and the EER (as used by NIST to evaluate SRE 2008 [11] and 2010 [16]), from left to right.

cients with cepstral mean subtraction, and their delta, double delta coefficients, producing a 60-dimensional feature vector. The resulting features are mean- and variance-normalized over the utterance. The feature vectors are modeled by a 1024-component, gender-independent GMM. We use a full Joint Factor Analysis model (JFA) in which 600 eigenvoices are trained and 250 eigenchannels are trained separately for telephone and interview data and are concatenated. The diagonal term is trained with the same data as used to train the speaker factors. Scores are normalized using gender-dependent ZTnorm, resulting in an EER of 1.65%, an old DCF of 0.073, and a new DCF of 0.42.

### 3.4 Fusion

Fusion results are obtained using a cross-validation paradigm. To this end, the complete set of speakers is split into two disjoint sets. The trials involving only speakers from each of these sets are then selected. In the process, half of the impostor trials (those corresponding to one speaker from one set and another speaker from the other set) are discarded. The fusion parameters are then trained using standard linear logistic regression on one of the sets and then applied to the other set, and conversely. The results shown in Table 1 are computed on the concatenation of these two sets. The fusion results show that the SNERF systems result in larger and more consistent gains over the baseline. This justifies using the SNERF features over the simpler polynomial features. Further, even though both SNERF systems give somewhat similar gains in combination, the proposed modeling technique should be more robust to noisy conditions and other types of variabilities, because the SNERF-SVM approach does not implement any kind of session variability compensation.

## 4 CONCLUSION

We have proposed a technique for modeling complex prosodic features, such as SNERFs, using a multinomial subspace model for feature extraction and probabilistic linear discriminant analysis for session variability compensation. The proposed system achieves more than 20% relative improvement with respect to the current prosodic systems on EER and old DCF metrics. An interesting finding is that the large gains from the proposed modeling technique decrease as the cost metric moves toward the low false acceptance region. In fact, at the recently introduced new DCF metric, which corresponds to very low false acceptance rates, both SNERF systems perform similarly. Comparing the performance of the polynomial prosodic features to the SNERFs, we see that SNERFs greatly outperform the simpler features. This behavior requires further investigation to understand whether it is due to the difference in the nature of the features, to the new modeling technique, or to both factors. Although SNERFs cannot be modeled with JFA, polynomial features could be modeled using the proposed MSM/PLDA technique. However, initial results in this direction did not show gains with respect to JFA modeling for these features.

In the future, we plan to investigate the performance of prosodic systems on diverse channel conditions and for different speech styles (interview conversations and telephone calls recorded over microphones other than telephone handsets). Further investigation is also needed to understand the influence of the subspace size. Finally, we plan to explore the use of heavy-tailed distributions in PLDA [14], which has been shown to give significant improvements for acoustic systems.

### 5 References

[1] D. Reynolds *et al.*, “The SuperSID project: Exploiting high-level information for high-accuracy,” in *in Proc. International Conference on Audio, Speech, and Signal Processing, Hong Kong*, 2003, pp. 784–787.

[2] ——, “The 2004 MIT lincoln laboratory speaker recognition system,” pp. 177 – 180, 2005.

[3] S. S. Kajarekar *et al.*, “SRIs 2004 nist speaker recognition evaluation system,” in *in Proc. ICASSP*, 2005, pp. 173–176.

[4] E. Shriberg *et al.*, “Modeling prosodic feature sequences for speaker recognition,” *Speech Communication*, Jan 2005.

[5] L. Ferrer *et al.*, “Parameterization of prosodic feature distributions for SVM modeling in speaker recognition,” *Proc. ICASSP, Taipei*, vol. 4, pp. 233–236, 2007.

[6] P. Kenny *et al.*, “A study of inter-speaker variability in speaker verification,” *IEEE Trans. Audio*, Jan 2008.

[7] N. Dehak *et al.*, “Modeling prosodic features with joint factor analysis for speaker verification,” *Audio, Speech and Language Processing*, Jan 2007.

[8] M. Kockmann *et al.*, “Investigations into prosodic syllable contour features for speaker recognition,” *Proc. of ICASSP, Dallas*, Sep 2010.

[9] L. Ferrer *et al.*, “A comparison of approaches for modeling prosodic features in speaker recognition,” in *Proc. ICASSP, Dallas*, 2010.

[10] M. Kockmann *et al.*, “Prosodic speaker verification using subspace multinomial models with intersession compensation,” in *Proc. Interspeech, Tokyo*, 2010.

[11] NIST, “The NIST year 2008 speaker recognition evaluation plan,” pp. 1–10, Apr 2008.

[12] S. Kajarekar *et al.*, “Modeling NERFs for speaker recognition,” in *Proc. Odyssey, Toledo*, 2004, pp. 51–56.

[13] S. J. D. Prince, “Probabilistic linear discriminant analysis for inferences about identity,” in *ICCV*, 2007.

[14] P. Kenny, “Bayesian speaker verification with heavy tailed priors,” in *Keynote presentation, Odyssey*, 2010.

[15] L. Burget *et al.*, “Discriminatively trained probabilistic linear discriminant analysis for speaker verification,” in *ICASSP*, 2011.

[16] NIST, “The NIST year 2010 speaker recognition evaluation plan,” 2010. [Online]. Available: http://www.itl.nist.gov/iad/mig//tests/sre/2010

[17] A. Solomonoff *et al.*, “Channel compensation for SVM speaker recognition,” in *Odyssey*, 2004, pp. 57–62.

# Regularized Subspace n-Gram Model for Phonotactic iVector Extraction

*Mehdi Soufifar*[1,2], *Lukáš Burget*[1], *Oldřich Plchot*[1], *Sandro Cumani*[1,3], *Jan "Honza" Černocký*[1]

[1] Brno University of Technology, BUT Speech@FIT and IT4I Centre of Excellence, Czech Republic
[2] Department of Electronics and Telecommunications, NTNU, Trondheim, Norway
[3] Politecnico di Torino, Italy

qsoufifar@stud.fit.vutbr.cz, {burget,cumani,cernocky,iplchot}@fit.vutbr.cz

## Abstract

Phonotactic language identification (LID) by means of n-gram statistics and discriminative classifiers is a popular approach for the LID problem. Low-dimensional representation of the n-gram statistics leads to the use of more diverse and efficient machine learning techniques in the LID. Recently, we proposed phototactic iVector as a low-dimensional representation of the n-gram statistics. In this work, an enhanced modeling of the n-gram probabilities along with regularized parameter estimation is proposed. The proposed model consistently improves the LID system performance over all conditions up to 15% relative to the previous state of the art system. The new model also alleviates memory requirement of the iVector extraction and helps to speed up subspace training. Results are presented in terms of $C_{avg}$ over NIST LRE2009 evaluation set.

**Index Terms**: Language identification, Subspace modeling, Subspace multinomial model

## 1. Introduction

State–of–the–art approaches to language identification (LID) can be mainly divided into two main categories: phonotactic LID and acoustic LID [1]. The phonotactic approach comprises techniques that use linguistic abstraction in speech modeling, while acoustic models try to infer the language of an utterance by directly modeling the spectral content of the utterance. This paper focuses on the phonotactic approach.

A successful representation of the phonetic content of utterances are n-gram statistics, which are often used as features for different language classifiers. However, the huge size of n-gram statistics poses some serious limitations on the choice of the LID backend classifier. Many solutions have been proposed to deal with the problem of n-gram vectors dimensionality. In [2], discriminative selection of the n-grams was proposed to discard less relevant n-grams. Many other phonotactic LID systems use principal component analysis (PCA) to reduce the dimensionality of the n-gram vectors [3, 4, 5]. We recently proposed a feature extraction technique based on subspace modeling of multinomial distribution parameters [6], where we showed that our approach outperforms former state of the art techniques based on n-gram statistics. This technique is inspired by the idea of iVector in acoustic speaker identification (SID) [7], where a low–dimensional vector is used to represent an utterance–dependent GMM supervector. In our context, we use a low–dimensional vector to represent the parameters of an utterance–dependent n-gram model.

The iVector extraction procedure presented in our previous work [6] was based on simpler Subspace Multinominal Model (SMM), where we assumed that n-grams are independent events generated from a single multinomial distribution and iVectors were computed as to maximize the likelihood of the observed n-grams. While this approach allows to obtain good results, the corresponding objective function is not directly related to the likelihood of the observed phoneme sequences. This is because the n-grams observed in a phoneme sequence are not independent. Using an n-gram model, likelihood of a phoneme sequence can be calculated as a product of the conditional probabilities of the individual phonemes given their histories. Such likelihood function is maximized in order to extract phonotactic iVectors using the Subspace n-Gram Model (SnGM) proposed in this work.

We found SnGM to be prone to over-fitting especially for short sequences, where only few different n-grams were observed. This was also one of the reasons for the former use of SMM, which is more robust to over-fitting. We show that this problem can be mitigated using regularization applied for both the subspace training and iVector extraction, which results in the superior performance of the newly proposed SnGM technique.

The paper is organized as follows: Section 2 describes the multinomial subspace model and details the subspace training and iVector extraction procedure. Section 3 describes our experimental setup. and compares the proposed method with PCA–based techniques and the multinomial model in [6]. An analysis of the model parameters is given in Section 4. Experimental results are reported in Section 5 and conclusions are drawn in Section 6.

## 2. Subspace multinomial model

In phonotactic LID, every speech utterance is tokenized to a sequence of phoneme labels. The n-gram model assumes that the probability of observing a phoneme is dependent only on the $n-1$ previous observed tokens. The log–likelihood of a

This work was partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. D10PC20015, by Czech Ministry of Trade and Commerce project No. FR-TI1/034, and by European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070). Sandro Cumani was supported by The European Social Fund (ESF) in the project Support of Interdisciplinary Excellence Research Teams Establishment at BUT (CZ.1.07/2.3.00/30.0005).

sequence of phonemes $l_1 \dots l_M$ can therefore be computed as

$$\log P(l_1 l_2 l_3 ... l_M) = \sum_i \log P(l_i | l_{i-n+1} l_{i-n+2} \dots l_{i-1}) \tag{1}$$

In order to model the phoneme generation process, we assume that the conditional distribution of a phoneme $l$ given a history $h$ is a multinomial distribution with parameters $\phi_{hl}$, i.e.

$$\log P(l|h) = \log \phi_{hl}, \tag{2}$$

with $\phi_{hl} > 0$ and $\sum_l \phi_{hl} = 1$. The joint log–likelihood of a sequence of phonemes $l_1 \dots l_M$ can then be computed as

$$\log P(l_1 l_2 l_3 ... l_M) = \sum_i \log P(l_i | h_i) = \sum_i \log \phi_{h_i l_i}, \tag{3}$$

where $h_i = (l_{i-n+1} l_{i-n+2} \dots l_{i-1})$ denotes the history for the observed phoneme $l_i$. The $\nu_{hl}$ denotes number of times the n-gram $hl$ (i.e. phoneme $l$ with history $h$) appears in the phoneme sequence, we can rewrite (3) as

$$\log P(l_1 l_2 l_3 ... l_M) = \sum_h \sum_l \nu_{hl} \log \phi_{hl}. \tag{4}$$

It is worth noting the difference between (3) and the objective that was maximized to obtain iVectors in [6] as:

$$\sum_{i=1}^{M} \log P(h_i, l_i) = \sum_h \sum_l \nu_{hl} \log \hat{\phi}_{hl}, \tag{5}$$

where n-grams were assumed to be generated independently from a single multinomial distribution (i.e. $\sum_h \sum_l \hat{\phi}_{hl} = 1$). This objective allows to obtain good performance. However, the corresponding iVectors do not maximize the likelihood of the observed phoneme sequence. In the following, we show how to build a phonotactic iVector extractor where iVectors are estimated in order to maximize the likelihood of the observed phoneme sequences under the n-gram model assumptions.

Our first step towards the phonotactic iVector extractor is to make assumption that, phoneme sequence from each utterance $s$ was generated from an utterance–specific n-gram distribution. Next, we assume that, the parameters of the corresponding multinomial distributions $\phi_{hl}(s)$ can be represented as

$$\phi_{hl}(s) = \frac{\exp(m_{hl} + \mathbf{t}_{hl}\mathbf{w}(s))}{\sum_i \exp(m_{hi} + \mathbf{t}_{hi}\mathbf{w}(s))}, \tag{6}$$

where $m_{hl}$ is the log-probability of n-gram $hl$ calculated over all the training data, $\mathbf{t}_{hl}$ is a row of a low–rank rectangular matrix $\mathbf{T}$ and $\mathbf{w}(s)$ is utterance–specific low–dimensional vector, which can be seen as low–dimensional representation of the utterance–specific n-gram model. The parameters $\{m_{hl}\}$ and the matrix $\mathbf{T}$ are the parameters of the proposed SnGM. Given these parameters, $\mathbf{w}(s)$ maximizing log-likelihood in (3) can be taken as the phonotactic iVector representing the an utterance $s$. Before iVectors can be extracted, however, the SnGM parameters have to be trained on a set of training utterances. This is done in an iterative EM-like process alternating between maximum likelihood (ML) updates of vectors $\mathbf{w}(s)$ (one for each training utterance $s$) and ML updates of SnGM parameters.

In the case of standard GMM based iVectors, the utterance–dependent parameters similar to $\mathbf{w}(s)$ are treated as latent random variables with standard normal priors. The subspace parameters are then trained using standard EM algorithm, where the M-step integrates over the latent variable posterior distributions from the E-step. Unfortunately, calculation of posterior distribution for $\mathbf{w}(s)$ is intractable in the case of SnGM. Instead, SnGM parameters are updated using only $\mathbf{w}(s)$ point estimates, which can negatively affect the robustness of SnGM parameter estimation. To mitigate this problem, we propose to regularize the ML objective function using L2 regularization terms for both the subspace matrix $\mathbf{T}$ and the vectors $\mathbf{w}(s)$. This corresponds to imposing an isotropic Gaussian prior on both the SnGM parameters and $\mathbf{w}(s)$, and obtaining MAP rather than ML point estimates. This is in contrast to our previous work [6], where only ordinary ML estimates of SnGm parameters and iVectors were used. In order to train our model, we maximize the regularized likelihood function

$$\sum_{s=1}^{S} \sum_h \sum_l \nu_{hl}(s) \log \phi_{hl}(s) - \frac{1}{2}\lambda \left\|\mathbf{t}_{hl}\right\|^2 - \frac{1}{2}\lambda \|\mathbf{w}(\mathbf{s})\|^2), \tag{7}$$

where the sum extends over all $S$ training utterances. The term $\lambda$ is the regularization coefficient for both the model parameters $\mathbf{T}$ and for $\mathbf{w}(s)$. Notice that we should regularize both $\mathbf{T}$ and $\mathbf{w}$ since limiting magnitude of $\mathbf{T}$ without regularizing $\mathbf{w}$ would be compensated by a dynamic range increase in $\mathbf{w}$.

### 2.1. Parameter estimation

The model parameters $m_{hl}$ are shared for all utterances and can be initialized as the logarithm of the conditional probability of a phoneme given its history computed over all training utterances:

$$m_{hl} = \log \left( \frac{\sum_s \nu_{hl}(s)}{\sum_s \sum_i \nu_{hi}(s)} \right). \tag{8}$$

In the following, we assume that the terms $m_{hl}$ do not require retraining. In order to alternately maximize the objective function (7) with respect to $\mathbf{T}$ and $\mathbf{w}$, we adapt the approach proposed in [8]. For a fixed $\mathbf{T}$, Newton Raphson-like update of $\mathbf{w}(s)$ is given by:

$$\mathbf{w}(s)^{new} = \mathbf{w}(s) + \mathbf{H}^{-1}_{w(s)} \boldsymbol{\nabla}_{w(s)}, \tag{9}$$

where the $\boldsymbol{\nabla}_{w(s)}$ is the gradient of the objective function (7) with respect to $\mathbf{w}(s)$

$$\boldsymbol{\nabla}_{w(s)} = \sum_h \sum_l \mathbf{t}^T_{hl} (\nu_{hl}(s) - \phi^{old}_{hl}(s) \sum_i \nu_{hi}(s)) - \lambda \mathbf{w}(s), \tag{10}$$

where the terms $\phi^{old}_{hl}(s)$ are the model parameters computed from the current estimate of $\mathbf{w}(s)$. $\mathbf{H}_{w(s)}$ is an approximation to the Hessian matrix proposed in [8] as

$$\mathbf{H}_{(\mathbf{w}(s))} = \sum_h \sum_l \mathbf{t}^T_{hl} \mathbf{t}_{hl} \max(\nu_{hl}(s), \phi^{old}_{hl}(s) \sum_i \nu_{hi}(s)) - \lambda \mathbf{I}. \tag{11}$$

Similarly, to update the $\mathbf{T}$ matrix, we keep all $\mathbf{w}(s)$ fixed and update each row of $\mathbf{T}$ as

$$\mathbf{t}^{new}_{hl} = \mathbf{t}_{hl} + \boldsymbol{\nabla}_{t_{hl}} \mathbf{H}^{-1}_{hl}, \tag{12}$$

where $\boldsymbol{\nabla}_{t_{hl}}$ is the gradient of the objective function (7) with respect to the row $\mathbf{t}_{hl}$ of $\mathbf{T}$

$$\boldsymbol{\nabla}_{t_{hl}} = \sum_s (\nu_{hl}(s) - \phi^{old}_{hl}(s) \sum_i \nu_{hi}(s)) \mathbf{w}(s)^T - \lambda \mathbf{t}_{hl}, \tag{13}$$

and

$$\mathbf{H}_{t_{hl}} = \sum_s \max(\nu_{hl}(s), \phi_{hl}^{old}(s) \sum_i \nu_{hi}(s)) \mathbf{w}(s)\mathbf{w}(s)^T - \lambda \mathbf{I}. \tag{14}$$

Notice that in (13), since we only need the n-gram statistics corresponding to n-gram history $h$, there is no need to load the whole vector of n-gram statistics. This reduces memory overhead of the $\mathbf{T}$ matrix update. Moreover, update of $\mathbf{T}$ rows belonging to different histories are completely independent, which simplifies parallel estimation of $\mathbf{T}$.

In our experiments Matrix $\mathbf{T}$ is initialized with small random numbers. Update of $\mathbf{T}$ or $\mathbf{w}(\mathrm{s})$ may fail to increase the objective function in (7). In that case, we keep backtracking by halving the update step. In case the objective function did not improve after certain number of backtracking, we retain the value of $\mathbf{t}_{hl}$ or $\mathbf{w}(s)$ from the previous iteration. The iterative parameter estimation continues until the change in the objective function becomes negligible. Once the SnGM is trained and fixed, it can be used to extract iVectors from new utterances by iteratively applying $\mathbf{w}(s)$ update formulas (9)-(11).

# 3. Experimental setup

To keep the results comparable to previously reported ones in [6], we report performance of the system over NIST LRE2009. We briefly explain the system description and the tuning. Interested readers are referred to the corresponding detailed system description [9].

## 3.1. Data

The LRE09 task comprises 23 languages. The EVAL set contains telephone data and narrowband broadcast data. The training data is divided into two sets denoted as TRAIN and DEV, both of which comprises data from 23 languages corresponding to the target list of the NIST LRE09 task [10]. The TRAIN set is filtered in order to keep at most 500 utterances per language as proposed in [9], resulting in 9763 segments (345 hours of recording). This allows to have almost balanced amounts of training data per language, thus avoiding biasing the classifiers toward languages with lots of training data. The DEV set contains 38469 segments mainly from the previous NIST LRE tasks plus some extra longer segments from the standard conversational telephone speech (CTS) databases (CallFriend, Switchboard, etc.) and voice of America (VOA). The TRAIN and the DEV sets contain disjoint sets of speakers. The DEV set is used to tune parameters and score calibration in the backend. A full description of the used data is given in [9].

## 3.2. Vector of n-gram counts

The n-gram counts were extracted using the Brno university of technology (BUT) Hungarian phone recognizer, which is an ANN/HMM hybrid [11]. The Hungarian phoneme list contains 51 phonemes. We map short and long variations of similar phonemes to the same token, obtaining 33 phonemes. This results in $33^3 = 35937$ 3-grams. Since neither 2-grams nor 1-grams improved the system performance we use only 3-gram counts. The 3-gram expected counts are extracted from phone lattices generated by the Hungarian phone recognizer.

## 3.3. Back end

We showed in [12] that iVector normalization is necessary to good LID performance using phonotactic iVectors. For this work, after mean removal, length normalized iVectors are used to train 23 logistic regression (LR) classifiers in one-vs-all configuration using LIBLINEAR[1]. The scores generated by 23 LR classifiers are calibrated on DEV data by means of a linear generative model followed by a multi-class LR as described in [13].

Table 1: $C_{avg} \times 100$ for different systems on NIST LRE09 Evaluation task over 30s, 10s and 3s conditions.

| System | Reg. Coef. | 30s | 10s | 3s |
|---|---|---|---|---|
| PCA | - | 2.93 | 8.29 | 22.60 |
| SMM | - | 2.81 | 8.33 | 21.39 |
| SnGM | - | 2.68 | 8.63 | 23.15 |
| RSnGM | 0.01 | **2.52** | **7.06** | **19.11** |

# 4. Analysis of the model parameters

Optimizing the objective function in (7) with $L2$ regularizer can be seen as obtaining MAP point estimate of the model parameters $\mathbf{T}$ and $\mathbf{w}$ with Gaussian priors. In Figure 2, the histogram of 10 random dimensions of $\mathbf{w}$ over TRAIN set and histogram of 10 random rows of the matrix $\mathbf{T}$ are depicted. The $y$ axis in both cases is the frequency of the bin. It can be seen from Figure 2 that the values in case of $\mathbf{w}$ are Gaussian distributed, which confirms assumption of the Gaussian priors over $\mathbf{w}$ vectors is appropriate. On the other hand, in the case of $\mathbf{T}$ rows, values seem to be Laplace distributed. This is mainly because the subspace matrix $\mathbf{T}$ is expanding the iVector space to the sparse original space of n-gram log-probabilities. Intuitively, this suggests use of an $L1$ regularizer that corresponds to the assumption of Laplace prior over estimation of the $\mathbf{T}$ matrix.

# 5. System evaluation & analysis

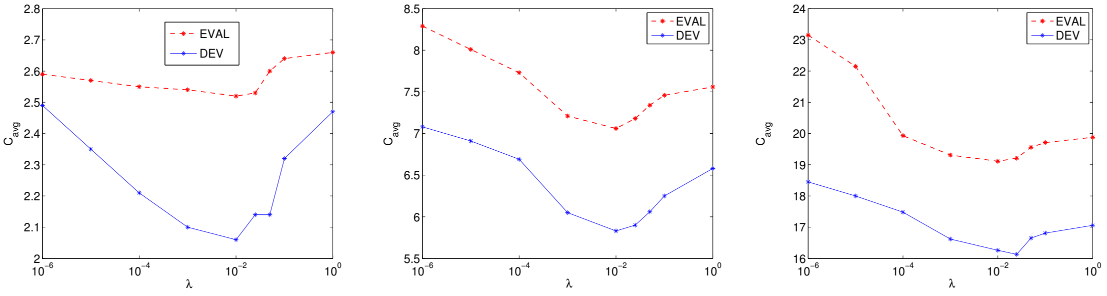
We showed in [12] that 600 is a reasonable choice for the subspace dimension over LRE2009 task. A 600 dimensional subspace and 5 iterations of parameter estimation is used since the value of the objective function over TRAIN set seems to converge after 4 iterations.

In Table 1, performance of the proposed SnGM (without regularization) is compared with subspace multinomial model (SMM) [6] and PCA-based feature extraction that is developed according to the recipe from [4]. The PCA system was widely used by the participants of NIST LRE11 as a phonotactic state of the art system. Aside from marginal degradation for $10s$ condition, the SMM outperforms PCA.

The SnGM system shows notable improvement over the baseline for the 30s condition. However, it also shows performance degradation over shorter conditions. We also noticed big dynamic range for the iVectors corresponding to the short utterances. Intuitively, for utterances with only few n-grams, there can be subspace basis (columns of $\mathbf{T}$) that do not (significantly) affect multinomial distributions corresponding to the seen histories. When estimating iVectors, its coefficients corresponding to such basis can take "arbitrary" values without affecting the likelihood of the observed n-grams. Note that SMM with single multinomial distribution does not suffer from this problem, and as such can be more robust to over-fitting.

To address the problem with over-fitting, we proposed SnGM with regularized parameter estimation (RSnGM). We

[1] http://www.csie.ntu.edu.tw/ cjlin/liblinear

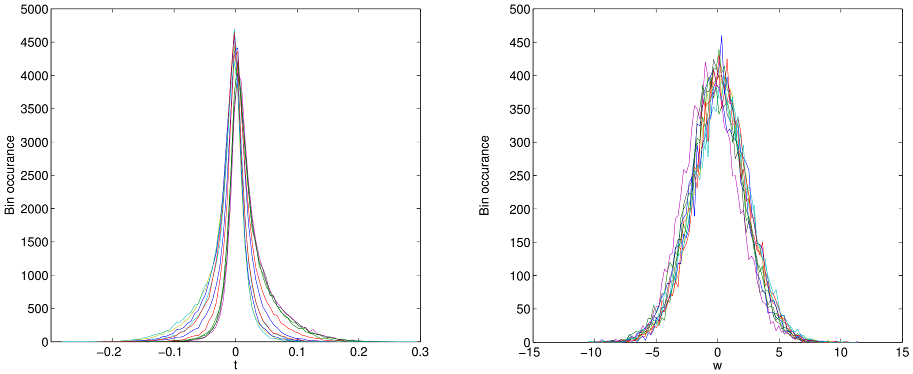


(a) 30 seconds

(b) 10 seconds

(c) 3 seconds

Figure 1: Effect of $\lambda$ on $C_{avg} \times 100$ over DEV and EVL set for 30s, 10s and 3s conditions on NIST LRE09



(a) Distribution of values in $\mathbf{T}$ rows

(b) Distribution of $\mathbf{w}$ over TRAIN set

Figure 2: Distribution of the values in the model parameters $\mathbf{T}$ and $\mathbf{w}$

use a grid search with logarithmic scale to tune the regularizer coefficient $\lambda$. This is depicted in Figure 1. The $\lambda$ value is tuned over the DEV set. Figure 1 shows that the best LID performance in terms of the $C_{avg}$ over DEV set is obtained with $\lambda = 0.01$. We also depicted the system performance on the held out EVAL set to study generalization of the $\lambda$ tuning to other unseen data. Interestingly, Figure 1 shows that the tuning of $\lambda$ over the DEV set generalizes well to the LRE09 EVAL set since the best performance on the NIST LRE09 EVAL set over all conditions are also obtained with $\lambda = 0.01$.

Table 1 shows effect of the regularized parameter estimation on the overall system performance. Results show that the RSnGM system shows significant improvement over the other state of the art systems.

## 6. Conclusion & future works

We proposed an enhanced phonotactic iVector extraction model over the n-gram counts. In the first step, a subspace n-gram model is proposed to model conditional n-gram probabilities. Modeling different 3-gram histories with separated multinomial distributions shows promising results for the long condition however, we observed model over-fitting for the short duration conditions.

Dealing with the model over-fitting problem, a regularized parameter estimation is proposed. Comparing the effect of the regularized and non-regularized parameter estimation on the overall system performance shows that the regularized parameter estimation is necessary to avoid over fitting of the subspace to the TRAIN set particularly for the short utterances. The proposed regularized subspace n-gram model shows consistent and significant improvement compared to the state of the art phonotactic systems as our baseline over all conditions. To the very best knowledge of the author, this is the best result reported on this task.

The Subspace n-gram model also reduces memory requirement for the parameter estimation and simplifies parallel parameter estimation that leads to a faster model training.

Our experiment with the proposed model shows importance of the numerical optimization during the parameter estimation. Since the $\mathbf{T}$ matrix is expanding iVector to a huge sparse space of the n-gram log-probabilities, use of an *L1* regularizer for estimating the $\mathbf{T}$ matrix may give us a better subspace model and will be explored in future.

## 7. References

[1] M. Zissman, “Comparison of four approaches to automatic language identification of telephone speech,” *Speech and Audio Processing, IEEE Transactions on*, vol. 4, no. 1, p. 31, 1996.

[2] F. Richardson and W. Campbell, “Language recognition with discriminative keyword selection,” in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, 2008, pp. 4145–4148.

[3] H. Li, B. Ma, and C.-H. Lee, “A vector space modeling approach to spoken language identification,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, pp. 271–284, 2007.

[4] T. Mikolov, O. Plchot, O. Glembek, P. Matějka, L. Burget, and J. Černocký, “Pca-based feature extraction for phonotactic language recognition,” in *Proc. Odyssey 2010 - The Speaker and Language Recognition Workshop*, 2010, pp. 251–255.

[5] S. M. Siniscalchi, J. Reed, and T. Svendsen, “Universal attribute characterization of spoken languages for automatic spoken language recognition,” *Computer Speech & Language Language*, 2013.

[6] M. Soufifar, M. Kockmann, L. Burget, O. Plchot, O. Glembek, and T. Svendsen, “ivector approach to phonotactic language recognition,” in *Proceedings of Interspeech 2011*, Florence, IT, 2011.

[7] N. Dehak, P. Kenny, R. eda Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Trans. Audio, Speech and Language Processing*, pp. 1–23, Jul 2009.

[8] D. Povey, L. Burget, M. Agarwal, P. Akyazi, A. Ghoshal, O. Glembek, K. N. Goel, M. Karafiát, A. Rastrow, R. Rose, P. Schwarz, and S. Thomas, “The subspace gaussian mixture model-a structured model for speech recognition,” *Computer Speech & Language*, vol. 25, no. 2, pp. 404–439, April 2011.

[9] N. Brümmer, L. Burget, O. Glembek, V. Hubeika, Z. Jančík, M. Karafiát, P. Matějka, T. Mikolov, O. Plchot, and A. Strasheim. But-agnitio system description for nist language recognition evaluation 2009. [Online]. Available: http://www.fit.vutbr.cz/research/groups/speech/publi/2009/brummer_BUT_AGNITIO_LRE09_SYSD.pdf

[10] “The 2009 NIST Language Recognition Evaluation Plan (LRE09),” http://www.itl.nist.gov/iad/mig/tests/lre/2009/LRE09_EvalPlan_v6.pdf.

[11] P. Schwarz, P. Matejka, and J. Cernocky, “Hierarchical structures of neural networks for phoneme recognition,” *Proceedings of ICASSP 2006, Toulouse*, pp. 325–328, Mar 2006.

[12] M. Soufifar, S. Cumani, L. Burget, and J. Černocký, “Discriminative classifiers for phonotactic language recognition with ivectors,” in *Proc. International Conference on Acoustics, Speech, and Signal Processing 2012*. Kyoto, Japan: IEEE Signal Processing Society, 2012, pp. 4853–4857.

[13] Z. Jančík, O. Plchot, N. Brummer, L. Burget, O. Glembek, V. Hubeika, M. Karafiát, P. Matějka, T. Mikolov, A. Strasheim, and J. Černocký, “Data selection and calibration issues in automatic language recognition - investigation with BUT-AGNITIO NIST LRE 2009 system,” in *Proc. Odyssey 2010 - The Speaker and Language Recognition Workshop*. Brno, CZ: International Speech Communication Association, 2010, pp. 215–221.

# Chapter 6

# Extensions of the i-vectors concept

This chapter deals with different extensions and modifications of the model for i-vector extraction. A simplified i-vector extraction model is proposed (section 6.1) in order to facilitate implementations of i-vector extraction into resource limited embedded devices. Discriminative training of such simplified model is proposed (section 6.2) to compensate for the performance loss introduced by the approximations used. Finally, extensions of i-vector extractor robust to additive background noise are proposed in papers from sections 6.3 and 6.4.

# SIMPLIFICATION AND OPTIMIZATION OF I-VECTOR EXTRACTION

*Ondřej Glembek*[1], *Lukáš Burget*[1], *Pavel Matějka*[1], *Martin Karafiát*[1], *Patrick Kenny*[2]

[1]Speech@FIT group, Brno University of Technology, Czech Republic
[2]Centre de Recherche Informatique de Montréal (CRIM), Montréal, Canada
{glembek,burget,matejkap,karafiat}@fit.vutbr.cz,
{patrick.kenny}@crim.ca

## ABSTRACT

This paper introduces some simplifications to the i-vector speaker recognition systems. I-vector extraction as well as training of the i-vector extractor can be an expensive task both in terms of memory and speed. Under certain assumptions, the formulas for i-vector extraction—also used in i-vector extractor training—can be simplified and lead to a faster and memory more efficient code. The first assumption is that the GMM component alignment is constant across utterances and is given by the UBM GMM weights. The second assumption is that the i-vector extractor matrix can be linearly transformed so that its per-Gaussian components are orthogonal. We use PCA and HLDA to estimate this transform.

***Index Terms*—** speaker recognition, i-vectors, Joint Factor Analysis, PCA, HLDA

## 1. INTRODUCTION

The i-vector systems have become the state-of-the-art technique in the speaker verification field [1]. They provide an elegant way of reducing the large-dimensional input data to a small-dimensional feature vector while retaining most of the relevant information. The technique was originally inspired by Joint Factor Analysis framework introduced in [2, 3].

The computational requirements for training the i-vector systems and estimating the i-vectors, however, are too high for certain types of applications. In this paper we propose simplifications to the original i-vector extraction and training schemes, which would dramatically decrease their complexity while retaining the recognition performance.

Our main motivation was running robust speaker verification systems on small scale devices such as mobile phones, as well as speeding up the process of speaker verification in real-time systems.

This paper is organized as follows: Section 2 introduces theoretical background of i-vector extraction and training of the i-vector extractor, Sections 3 and 4 introduce the proposed methods for i-vector extraction, Section 5 describes the experimental setup, Section 6 presents the recognition, speed, and memory performance, and Section 7 concludes the paper.

## 2. THEORETICAL BACKGROUND

Let us first state the motivation for the i-vectors. The main idea is that the speaker- and channel-dependent GMM supervector $\mathbf{s}$ can be modeled as:

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{w} \tag{1}$$

where $\mathbf{m}$ is the UBM GMM mean supervector, $\mathbf{T}$ is a low-rank matrix representing $M$ bases spanning subspace with important variability in the mean supervector space, and $\mathbf{w}$ is a standard normal distributed vector of size $M$.

For each observation $\mathcal{X}$, the aim is to estimate the parameters of the posterior probability of $\mathbf{w}$:

$$\mathrm{p}(\mathbf{w}|\mathcal{X}) = \mathcal{N}(\mathbf{w}; \mathbf{w}_{\mathcal{X}}, \mathbf{L}_{\mathcal{X}}^{-1}) \tag{2}$$

The i-vector is the MAP point estimate of the variable $\mathbf{w}$, i.e. the mean $\mathbf{w}_{\mathcal{X}}$ of the posterior distribution $\mathrm{p}(\mathbf{w}|\mathcal{X})$. It maps most of the relevant information from a variable-length observation $\mathcal{X}$ to a fixed- (small-) dimensional vector. $\mathbf{T}$ is referred to as the i-vector extractor.

### 2.1. Data

The input data for the observation $\mathcal{X}$ is given as a set of *zero-* and *first-order statistics* — $\mathbf{n}_{\mathcal{X}}$ and $\mathbf{f}_{\mathcal{X}}$. These are extracted from $F$ dimensional features using a GMM UBM with $C$ mixture components, defined by a mean supervector $\mathbf{m}$, component covariance matrices $\mathbf{\Sigma}^{(c)}$, and a vector of mixture weights $\boldsymbol{\omega}$. For each Gaussian component $c$, the statistics are given respectively as:

$$N_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \tag{3}$$

$$\mathbf{f}_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \mathbf{o}_t \tag{4}$$

where $\mathbf{o}_t$ is the feature vector in time $t$, and $\gamma_t^{(c)}$ is its occupation probability. The complete zero- and first-order statistics supervectors are $\mathbf{f}_{\mathcal{X}} = \left(\mathbf{f}_{\mathcal{X}}^{(1)\prime}, \ldots, \mathbf{f}_{\mathcal{X}}^{(C)\prime}\right)^{\prime}$, and $\mathbf{n}_{\mathcal{X}} = \left(N_{\mathcal{X}}^{(1)}, \ldots, N_{\mathcal{X}}^{(C)}\right)^{\prime}$.

For convenience, we *center* the first order statistics around the UBM means, which allows us to treat the UBM means effectively as a vector of zeros:

$$\begin{aligned}\mathbf{f}_{\mathcal{X}}^{(c)} &\leftarrow \mathbf{f}_{\mathcal{X}}^{(c)} - N_{\mathcal{X}}^{(c)}\mathbf{m}^{(c)} \\ \mathbf{m}^{(c)} &\leftarrow \mathbf{0}\end{aligned}$$

Similarly, we "normalize" the first-order statistics and the matrix $\mathbf{T}$ by the UBM covariances, which again allows us to treat the UBM covariances as an identity matrix[1]:

$$\begin{aligned}\mathbf{f}_{\mathcal{X}}^{(c)} &\leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}}\mathbf{f}_{\mathcal{X}}^{(c)} \\ \mathbf{T}^{(c)} &\leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}}\mathbf{T}^{(c)} \\ \mathbf{\Sigma}^{(c)} &\leftarrow \mathbf{I}\end{aligned}$$

[1] Part of the factor estimation is a computation of $\mathbf{T}'\mathbf{\Sigma}^{-1}\mathbf{f}$, where the decomposed $\mathbf{\Sigma}^{-1}$ can be projected to the neigboring terms, see [2] for detailed formulae.

where $\mathbf{\Sigma}^{(c)-\frac{1}{2}}$ is a Cholesky decomposition of an inverse of $\mathbf{\Sigma}^{(c)}$, and $\mathbf{T}^{(c)}$ is a $F \times M$ sub-matrix of $\mathbf{T}$ corresponding to the $c$ mixture component such that $\mathbf{T} = \left(\mathbf{T}^{(1)'}, \ldots, \mathbf{T}^{(C)'}\right)'$.

### 2.2. Parameter Estimation

As described in [2] and with the data transforms from previous section, for an observation $\mathcal{X}$, the corresponding i-vector is computed as a point estimate:

$$\mathbf{w}_{\mathcal{X}} = \mathbf{L}_{\mathcal{X}}^{-1}\mathbf{T}'\mathbf{f}_{\mathcal{X}} \tag{5}$$

where $\mathbf{L}$ is the precision matrix of the posterior distribution, computed as:

$$\mathbf{L}_{\mathcal{X}} = \mathbf{I} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)}\mathbf{T}^{(c)'}\mathbf{T}^{(c)} \tag{6}$$

The computational complexity of the whole estimation for one observation is $O(CFM + CM^2 + M^3)$. The first term represents the $\mathbf{T}'\mathbf{f}_{\mathcal{X}}$ multiplication. The second term represents the sum in (6) and includes the multiplication of $\mathbf{L}_{\mathcal{X}}^{-1}$ with a vector. The third term represents the matrix inversion.

The memory complexity of the estimation is $O(CFM+CM^2)$. The first term represents the storage of all the input variables in (5), and the second term represents the pre-computed matrices in the sum of (6).

Note that the computation complexity grows quadratically with $M$ in the sum of (6), and linearily with $C$. This becomes the bottleneck in the i-vector computation, resulting in high memory and CPU demands.

### 2.3. Model Training

Model hyper-parameters $\mathbf{T}$ are estimated using the same EM algorithm as in case of JFA [2]. Note that our algorithm makes use of an additional *minimum divergence* update step [3, 4], which yields a quicker convergence, but is not described here.

In the E step, the following accumulators are collected using all training observations $i$:

$$\mathbf{C} = \sum_i \mathbf{f}_i\mathbf{w}_i' \tag{7}$$

$$\mathbf{A}^{(c)} = \sum_i N_i^{(c)}\left(\mathbf{L}_i^{-1} + \mathbf{w}_i\mathbf{w}_i'\right) \tag{8}$$

where $\mathbf{w}_i$ and $\mathbf{L}_i$ are the estimates from (5) and (6) for observation $i$. The M step update is given as follows:

$$\mathbf{T}^{(c)} = \mathbf{C}\mathbf{A}^{(c)^{-1}} \tag{9}$$

## 3. SIMPLIFICATION 1: CONSTANT GMM COMPONENT ALIGNMENT

In this method, we apply the assumption that the GMM component alignment is constant across segments, i.e. the posterior occupation probabilities $\gamma^{(c)}$ in (3) are replaced by their prior probabilities represented by the UBM GMM weights. The new zero-order statistics are then:

$$\bar{N}_{\mathcal{X}}^{(c)} = \omega^{(c)}N_{\mathcal{X}} \tag{10}$$

where $\omega^{(c)}$ is the GMM UBM weight of component $c$, and $N_{\mathcal{X}} = \sum_{j=1}^{C} N_{\mathcal{X}}^{(j)}$. Substituting $N_{\mathcal{X}}^{(c)}$ in (6) by $\bar{N}_{\mathcal{X}}^{(c)}$ from (10), we get

$$\bar{\mathbf{L}}_{\mathcal{X}} = \mathbf{I} + N_{\mathcal{X}}\mathbf{W} \tag{11}$$

where

$$\mathbf{W} = \sum_{c=1}^{C} \omega^{(c)}\mathbf{T}^{(c)'}\mathbf{T}^{(c)} \tag{12}$$

Exploiting this simplification in the i-vector extractor training can be done at two stages: substituting $\mathbf{L}_i$ in (8) by (11), and substituting $N_i^{(c)}$ in (8) by (10). Based on our experiments, only the former turned out to be effective, therefore we will not report any results with the latter one.

Note that $\mathbf{W}$ in (12) is independent of data and can be pre-computed. Its resulting size is $M \times M$ yielding faster computation and less memory demands. The computational copmlexity of this algorithm reduces to $O(CFM + M^3)$ with the dominating inversion step. The memory complexity reduces to $O(CFM + M^2)$.

## 4. SIMPLIFICATION 2: I-VECTOR EXTRACTOR ORTHOGONALIZATION

Let us assume, that we can find a linear (orthogonal) transformation $\mathbf{G}$ which would orthogonalize all individual per-component sub-matrices $\mathbf{T}^{(c)}$. Orthogonalizing $\mathbf{T}$ would diagonalize $\mathbf{L}_{\mathcal{X}}$, which would need to be rotated back using $\mathbf{G}$. We can then express (6) as

$$\mathbf{L}_{\mathcal{X}} = \mathbf{G}^{(-1)'}\hat{\mathbf{L}}_{\mathcal{X}}\mathbf{G}^{-1} \tag{13}$$

where

$$\hat{\mathbf{L}}_{\mathcal{X}} = \mathbf{G}'\mathbf{G} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)}\mathbf{G}'\mathbf{T}^{(c)'}\mathbf{T}^{(c)}\mathbf{G} \tag{14}$$

Assuming that $\hat{\mathbf{L}}_{\mathcal{X}}$ is diagonal, we can rewrite it as

$$\hat{\mathbf{L}}_{\mathcal{X}} = \mathrm{Diag}\left(\mathrm{diag}(\mathbf{G}'\mathbf{G}) + \mathbf{V}\mathbf{n}_{\mathcal{X}}\right) \tag{15}$$

where $\mathbf{V}$ is a $M \times C$ matrix whose $c$th column is $\mathrm{diag}(\mathbf{G}'\mathbf{T}^{(c)'}\mathbf{T}^{(c)}\mathbf{G})$. $\mathrm{Diag}(\cdot)$ maps a vector to a diagonal matrix, while $\mathrm{diag}(\cdot)$ maps a matrix diagonal to a vector. Combining (13) and (5), we get

$$\hat{\mathbf{w}}_{\mathcal{X}} = \mathbf{G}\hat{\mathbf{L}}_{\mathcal{X}}^{-1}\mathbf{G}'\mathbf{T}'\mathbf{f}_{\mathcal{X}} \tag{16}$$

The computational complexity of this approach is $O(CFM)$ as we can effectively simplify the matrix inversion to a vector element-wise inversion. The memory complexity is $O(CFM + M^2 + CM)$, where $M^2$ represents the extra diagonalization matrix $\mathbf{G}$, and $CM$ represents $\mathbf{V}$ from (15).

The task is to estimate the orthogonalization matrix $\mathbf{G}$. Let us take a look at two approaches we investigated:

### 4.1. Eigen-decomposition

Let $\mathbf{W}$ be the weighted average per-component covariance matrix from (12). We assume $\mathbf{W}$ to be a full-rank matrix with $M$ linearly independent eigenvectors. Then $\mathbf{W}$ can be factorized as

$$\mathbf{W} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1} \tag{17}$$

where $\mathbf{Q}$ is a square $M \times M$ matrix whose $i$th column is the eigenvector $\mathbf{q}_i$ of $\mathbf{W}$ and $\mathbf{\Lambda}$ is a diagonal matrix whose diagonal elements are the corresponding eigenvalues. Matrix $\mathbf{Q}$ clearly orthogonalizes the space given by $\mathbf{W}$, therefore we can set $\mathbf{G} = \mathbf{Q}$.

### 4.2. Heteroscedastic Linear Discriminant Analysis

If the average covariance matrix $\mathbf{W}$ from (12) is close to diagonal, then the eigen-decomposition is not effective in diagonalizing the per-component covariances.

HLDA is a supervised method, which allows us to derive such projection that best de-correlates features associated with each particular class (maximum likelihood linear transformation for diagonal covariance modeling [5]). An efficient iterative algorithm [6] was used in our experiments to estimate matrix $\mathbf{G}$. In our task, the classes were defined as Gaussian mixture components. The within-class covariance matrices were given by $\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}$, and the occupation counts were provided as the mixture weights $\omega^{(c)}$.

Note that the well known Linear Discriminant Analysis (LDA) can be seen as special case of HLDA, where it is assumed that covariance matrices of all classes are the same.

## 5. EXPERIMENTAL SETUP

### 5.1. Feature Extraction

In our experiments, we used cepstral features, extracted using a 25 ms Hamming window. 19 Mel frequency cepstral coefficients together with log-energy were calculated every 10 ms. This 20-dimensional feature vector was subjected to short time mean and variance normalization using a 3s sliding window. Delta and double delta coefficients were then calculated using a 5-frame window giving 60-dimensional feature vectors.

Segmentation was based on the BUT Hungarian phoneme recognizer and relative average energy thresholding. Also, short segments were pruned out, after which the speech segments were merged together.

### 5.2. System Training

One gender-independent universal background model was represented as a diagonal covariance, 2048-component GMM. It was trained using LDC releases of Switchboard II, Phases 2 and 3; switchboard Cellular, Parts 1 and 2 and NIST 2004-2005 SRE.

One (gender-dependent) i-vector extractor was trained on the female part of the following telephone data: NIST SRE 2004, NIST SRE 2005, NIST SRE 2006, Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, Fisher English Parts 1 and 2 giving 8396 female speaker in 1463 hours of speech, and 6168 male speakers in 1098 hours of speech (both after voice activity detection).

Originally, 400 dimensional i-vector extractor was chosen as a reference. As mentioned later, training of the 800 dimensional system got feasible using one of the proposed methods. We trained such system to demonstrate the potentials of the proposed methods.

### 5.3. Scoring and Normalization

The same technique as in [1] was used. The extracted i-vectors were scaled down using an LDA matrix to 200 dimensions, and further normalized by a within-class covariance matrix. Both of these matrices were gender-dependent and were estimated on the same data as the i-vector extractor, except the Fisher data was excluded, resulting in 1684 female speakers in 715 hours of speech and 1270 male speakers in 537 hours of speech.

Cosine distance of the two input vectors was used as the raw score:

$$\operatorname{score}\left(\mathbf{w}_{\text{target}}, \mathbf{w}_{\text{test}}\right) = \frac{\left\langle \mathbf{w}_{\text{target}}, \mathbf{w}_{\text{test}} \right\rangle}{\|\mathbf{w}_{\text{target}}\| \|\mathbf{w}_{\text{test}}\|} \qquad (18)$$

The cosine distance scores were normalized using gender-dependent s-norm [7] with a cohort of 400 speakers having 2 utterances per speaker.

### 5.4. Test Setup

The results of our experiments are reported on the female part of the Condition 5 (telephone-telephone) of the NIST 2010 speaker recognition evaluation (SRE) dataset [8]. The recognition accuracy is given as a set of equal error rate (EER), and the normalized DCF as defined both in the NIST 2010 SRE task ($\mathrm{DCF}_{\mathrm{new}}$) and the previous SRE evaluations ($\mathrm{DCF}_{\mathrm{old}}$).

The speed and memory performance of i-vector extraction were tested on a set of 50 randomly chosen utterances from the MIXER05 database. The input data (given as a set of fixed-size zero- and first-order statistics) and all of the input parameters were included in the general memory requirements. The following algorithm-specific terms were pre-computed (thus not included in the reported times), and comprised in the algorithm-specific memory requirements:

- $\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}$ in (6)
- $\mathbf{W}$ in (12)
- $\mathbf{G}$ and $\mathbf{T}^{(c)}\mathbf{G}$ in (13) and (16), and $\mathbf{V}$ in (15)

The algorithms were tested in MATLAB (R2009b) 64-bit, running in a single thread and the default double-precision mode. The machine was an Intel(R) Xeon(R) CPU X5670 2.93GHz, with 36GB RAM.

## 6. RESULTS

In the following section, we will reference the systems according to the i-vector dimensionality and to the extraction method used. *Baseline* stands for the original method as in Sec. 2.2, and *simple 1* and *simple 2* reference to the proposed simplifications.

Table 1 summarizes the systems with respect to verification accuracy. Fig. 1 visualizes the different systems on a constellation plot. The "800 baseline" system is clearly the winner, however "800 simple 2 - HLDA" is a tight competitor to the "400 baseline".

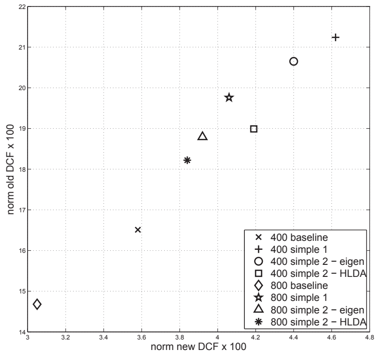


**Fig. 1**. Constellation plot of the individual systems

### 6.1. Speed and Memory

As described earlier in Sec. 5.4, the computation time does not include reading of the necessary data and pre-computation of some terms. The results are reported in Tab. 2. The dominating complexity of matrix inversion makes "simple 2" faster than "simple 1", as described in Sec. 3 and 4.

**Table 1**. *Comparison of the proposed i-vector extraction methods in terms of normalized DCFs and EER*

| | $\mathrm{DCF}_{\mathrm{new}}$ | $\mathrm{DCF}_{\mathrm{old}}$ | EER |
|---|---|---|---|
| 400 baseline | 0.5395 | 0.1651 | 3.58 |
| 400 simple 1 | 0.6664 | 0.2124 | 4.62 |
| 400 simple 2 - eigen | 0.6627 | 0.2065 | 4.40 |
| 400 simple 2 - HLDA | 0.6236 | 0.1899 | 4.19 |
| 800 baseline | 0.4956 | 0.1468 | 3.05 |
| 800 simple 1 | 0.6057 | 0.1976 | 4.06 |
| 800 simple 2 - eigen | 0.5414 | 0.1879 | 3.92 |
| 800 simple 2 - HLDA | 0.5694 | 0.1822 | 3.84 |

**Table 2**. *Comparison of the proposed i-vector extraction methods in processing speed.*

| | absolute [sec] | relative to 400 baseline |
|---|---|---|
| 400 baseline | 13.70 | 100.00% |
| 400 simple 1 | 1.01 | 7.37% |
| 400 simple 2 | 0.54 | 3.94% |
| 800 baseline | 65.75 | 480.00% |
| 800 simple 1 | 3.64 | 26.57% |
| 800 simple 2 | 1.11 | 8.10% |

Tab. 3 shows memory allocation for different systems. We see that for most of the current hardware configurations, the baseline systems could be a problem.

**Table 3**. *Comparison of the proposed i-vector extraction methods in memory allocation (in MB). The "constant" term depends on the i-vector dimensionality.*

| | constant | algorithm specific | total |
|---|---|---|---|
| 400 baseline | 422.96 | 2,500.00 | 2,923.00 |
| 400 simple 1 | " | 1.22 | 424.18 |
| 400 simple 2 | " | 7.47 | 430.43 |
| 800 baseline | 802.84 | 10,000.00 | 10,802.84 |
| 800 simple 1 | " | 4.88 | 807.83 |
| 800 simple 2 | " | 17.38 | 820.23 |

Note that prior to the scoring, WCCN and LDA dimensionality reduction are applied to the i-vectors (see Sec. 5.3). Projecting this linear transformation directly into the leftmost $\mathbf{G}$ of (16) could further decrease the complexity of the "simple 2" algorithm.

### 6.2. Simplification 1 in Training

While none of the simplifications had positive contribution to the test accuracy, the training phase simplification results in negligible accuracy changes while exploiting some of the speed and memory advantages as described in the previous section. Table 4 shows the difference.

Time and memory complexity of collecting the accumulators $\mathbf{A}$ from (8) is almost identical to the computation of $\mathbf{L}_{\mathcal{X}}$ in (6). The proposed method still keeps the same accumulator collection, however, avoiding the expensive computation of (6) decreases the E step time and memory complexity by a factor of 2.

**Table 4**. *Comparison of the proposed i-vector extractor training methods in terms of normalized DCFs and EER*

| | $\mathrm{DCF}_{\mathrm{new}}$ | $\mathrm{DCF}_{\mathrm{old}}$ | EER |
|---|---|---|---|
| 400 baseline | 0.5460 | 0.1722 | 3.40 |
| 400 simple 1 | 0.5376 | 0.1729 | 3.42 |

## 7. CONCLUSIONS

We managed to reduce the memory requirements and processing time for the i-vector extractor training so that higher dimensions can be now used while retaining the recognition accuracy. As for i-vector extraction, we managed to reduce the complexity of the algorithm with sacrificing little recognition accuracy, which makes this technique usable in small-scale devices.

As a practical result, Simplification 1 was used in the MOBIO project, when porting a speaker verification system on a mobile phone platform.

Not only we managed to scale down the complexity of the system in terms of real-world applications, but also we have prepared a set of simplified formulas which could potentially find use in a future research, such as discriminative training.

## 8. ACKNOWLEDGMENTS

The work was partly supported by European project MOBIO (FP7-214324), Grant Agency of Czech Republic project No. 102/08/0707, Czech Ministry of Education project No. MSM0021630528 and by BUT FIT grant No. FIT-10-S-2. Great part of the work was done at the BOSARIS workshop held at BUT in July 2010.

## 9. REFERENCES

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. PP, no. 99, 2010.

[2] P. Kenny, "Joint factor analysis of speaker and session variability : Theory and algorithms - technical report CRIM-06/08-13. Montreal, CRIM, 2005," 2005.

[3] P. Kenny, G. Boulianne, P. Oullet, and P. Dumouchel, "Joint factor analysis versus eigenchannes in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

[4] Niko Brümmer, "The EM algorithm and minimum divergence," Agnitio Labs Technical Report. Online: http://niko.brummer.googlepages.com/EMandMINDIV.pdf, Oct. 2009.

[5] N. Kumar, *Investigation of Silicon-Auditory Models and Generalization of Linear Discriminant Analysis for Improved Speech Recognition*, Ph.D. thesis, John Hopkins University, Baltimore, 1997.

[6] M.J.F. Gales, "Semi-tied covariance matrices for Hidden Markov Models," *IEEE Trans. Speech and Audio Processing*, vol. 7, pp. 272–281, 1999.

[7] N. Brümmer and A. Strasheim, "AGNITIO's speaker recognition system for EVALITA 2009," 2009.

[8] "National institute of standard and technology," http://www.nist.gov/speech/tests/spk/index.htm.

# Discriminatively Trained i-vector Extractor for Speaker Verification

Ondřej Glembek[1], Lukáš Burget[1], Niko Brümmer[2], Oldřich Plchot[1], Pavel Matějka[1]

[1]Speech@FIT group, Brno University of Technology, Czech Republic
[2]Agnitio, South Africa
{glembek,burget,matejkap}@fit.vutbr.cz,
niko.brummer@gmail.com

## Abstract

We propose a strategy for discriminative training of the i-vector extractor in speaker recognition. The original i-vector extractor training was based on the maximum-likelihood generative modeling, where the EM algorithm was used. In our approach, the i-vector extractor parameters are numerically optimized to minimize the discriminative cross-entropy error function. Two versions of the i-vector extraction are studied—the original approach as defined for Joint Factor Analysis, and the simplified version, where orthogonalization of the i-vector extractor matrix is performed.

**Index Terms**: speaker verification, i-vectors, PLDA, discriminative training

## 1. Introduction

Recently, systems based on i-vectors [1, 2] (extracted from cepstral features) have provided superior performance in speaker verification. The so-called i-vector is an information-rich low-dimensional fixed-length vector extracted from the feature sequence representing a speech segment (see Section 2 for details on i-vector extraction). A speaker verification score is produced by comparing two i-vectors corresponding to the segments in the verification trial. The function taking two i-vectors as an input and producing the corresponding verification score is designed to give the log-likelihood ratio between the "same-speaker" and "different-speaker" hypotheses. Best performance is currently obtained with Probabilistic Linear Discriminant Analysis (PLDA) [2]—a generative model that models i-vector distributions allowing for direct evaluation of the desired log-likelihood ratio verification score (see Section 2.4 for details).

In [3], it was shown that discriminatively training the PLDA parameters can lead to improvement in recognition performance. In this paper, we go deeper in the speaker recognition chain and we show that a similar discriminative training framework can be adopted for training the parameters of the i-vector extractor. We apply this technique in two kinds of i-vector extractor. In the first case, the traditional extraction—as proposed in [1]—is studied. It will be further referred to as the *full i-vector extractor*. Its parameters are given by a single matrix $\mathbf{T}$. In the second case, the simplified extraction (referred to as "Simplification 2" in [4]) is addressed. Its parameters are given by three matrices—$\mathbf{T}$, $\mathbf{G}$, and $\mathbf{V}$. It will be further referred to as the *simplified i-vector extractor*.

This paper is organized as follows: Section 2 introduces a theoretical background of the individual parts of the speaker recognition chain, Section 3 introduces the technique of discriminative training, Section 4 describes the experimental setup and results, and Section 5 concludes the paper.

## 2. Theoretical background

The i-vectors provide an elegant way of reducing large-dimensional input data to a small-dimensional feature vector while retaining most of the relevant information. The technique was originally inspired by Joint Factor Analysis (JFA) framework introduced in [5, 6].

The main idea is that the speaker- and channel-dependent Gaussian Mixture Model (GMM) supervector $\mathbf{s}$ can be modeled as:

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{w} \tag{1}$$

where $\mathbf{m}$ is the Universal Background Model (UBM) GMM mean supervector, $\mathbf{T}$ is a low-rank matrix representing $M$ bases spanning subspace with important variability in the mean supervector space, and $\mathbf{w}$ is a latent variable of size $M$ with standard normal distribution.

For each observation $\mathcal{X}$, the aim is to compute the parameters of the posterior probability of $\mathbf{w}$:

$$\mathrm{p}(\mathbf{w}|\mathcal{X}) = \mathcal{N}(\mathbf{w}; \mathbf{w}_{\mathcal{X}}, \mathbf{L}_{\mathcal{X}}^{-1}) \tag{2}$$

The i-vector $\boldsymbol{\phi}$ is the Maximum a Posteriori (MAP) point estimate of the variable $\mathbf{w}$, i.e., the mean $\mathbf{w}_{\mathcal{X}}$ of the posterior distribution $\mathrm{p}(\mathbf{w}|\mathcal{X})$. It maps most of the relevant information from a variable-length observation $\mathcal{X}$ to a fixed- (small-) dimensional vector. $\mathbf{L}_{\mathcal{X}}$ is the precision of the posterior distribution.

### 2.1. Sufficient statistics

The input data for the observation $\mathcal{X}$ is given as a set of *zero-* and *first-order statistics* — $\mathbf{n}_{\mathcal{X}}$ and $\mathbf{f}_{\mathcal{X}}$. These are extracted from $F$ dimensional features using a GMM UBM with $C$ mixture components, defined by a mean supervector $\mathbf{m}$, component covariance matrices $\boldsymbol{\Sigma}^{(c)}$, and a vector of mixture weights $\boldsymbol{\omega}$. For each Gaussian component $c$, the statistics are given respectively as

$$N_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \tag{3}$$

$$\mathbf{f}_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \mathbf{o}_t \tag{4}$$

BUT researchers carrying on this work were funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory (ARL). All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI, or the U. S. Government.

where $\mathbf{o}_t$ is the feature vector in time $t$, and $\gamma_t^{(c)}$ is its occupation probability. The complete zero- and first-order statistics supervectors are $\mathbf{f}_{\mathcal{X}} = \left(\mathbf{f}_{\mathcal{X}}^{(1)\prime}, \ldots, \mathbf{f}_{\mathcal{X}}^{(C)\prime}\right)'$, and $\mathbf{n}_{\mathcal{X}} = \left(N_{\mathcal{X}}^{(1)}, \ldots, N_{\mathcal{X}}^{(C)}\right)'$.

For convenience, we *center* the first-order statistics around the UBM means, which allows us to treat the UBM means effectively as a vector of zeros:

$$\begin{aligned} \mathbf{f}_{\mathcal{X}}^{(c)} &\leftarrow \mathbf{f}_{\mathcal{X}}^{(c)} - N_{\mathcal{X}}^{(c)}\mathbf{m}^{(c)} \\ \mathbf{m}^{(c)} &\leftarrow \mathbf{0} \end{aligned}$$

Similarly, we “normalize” the first-order statistics and the matrix $\mathbf{T}$ by the UBM covariances, which again allows us to treat the UBM covariances as an identity matrix:[1]

$$\begin{aligned} \mathbf{f}_{\mathcal{X}}^{(c)} &\leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}}\mathbf{f}_{\mathcal{X}}^{(c)} \\ \mathbf{T}^{(c)} &\leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}}\mathbf{T}^{(c)} \\ \mathbf{\Sigma}^{(c)} &\leftarrow \mathbf{I} \end{aligned}$$

where $\mathbf{\Sigma}^{(c)-\frac{1}{2}}$ is a Cholesky decomposition of an inverse of $\mathbf{\Sigma}^{(c)}$, and $\mathbf{T}^{(c)}$ is an $F \times M$ submatrix of $\mathbf{T}$ corresponding to the $c$ mixture component such that $\mathbf{T} = \left(\mathbf{T}^{(1)\prime}, \ldots, \mathbf{T}^{(C)\prime}\right)'$.

### 2.2. i-vector extraction

As described in [5] and with the data transforms from the previous section, for an observation $\mathcal{X}$, the corresponding i-vector is computed as a point estimate:

$$\boldsymbol{\phi}_{\mathcal{X}} = \mathbf{L}_{\mathcal{X}}^{-1}\mathbf{T}'\mathbf{f}_{\mathcal{X}} \tag{5}$$

where $\mathbf{L}$ is the precision matrix of the posterior distribution, computed as

$$\mathbf{L}_{\mathcal{X}} = \mathbf{I} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)}\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)} \tag{6}$$

### 2.3. i-vector extraction—simplified version

According to [4], the i-vector extraction can be simplified to reduce the computation complexity. Assuming there is a linear (orthogonal) transformation $\mathbf{G}$ that would orthogonalize all individual per-component submatrices $\mathbf{T}^{(c)}$, the i-vector extraction can be expressed as

$$\hat{\boldsymbol{\phi}}_{\mathcal{X}} = \mathbf{G}\hat{\mathbf{L}}_{\mathcal{X}}^{-1}\mathbf{G}'\mathbf{T}'\mathbf{f}_{\mathcal{X}} \tag{7}$$

where

$$\hat{\mathbf{L}}_{\mathcal{X}} = \mathrm{Diag}\left(\mathbf{I} + \mathbf{V}\mathbf{n}_{\mathcal{X}}\right) \tag{8}$$

where $\mathbf{V}$ is an $M \times C$ matrix whose $c$th column is $\mathrm{diag}(\mathbf{G}'\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}\mathbf{G})$. $\mathrm{Diag}(\cdot)$ maps a vector to a diagonal matrix.

### 2.4. PLDA

To facilitate comparison of i-vectors in a verification trial, we use a Probabilistic Linear Discriminant Analysis (PLDA) model [7, 2]. It can be seen as a special case of JFA with a single Gaussian component. Given a pair of i-vectors, PLDA allows to compute the log-likelihood for the same-speaker hypothesis and for the different-speaker hypothesis. One can directly evaluate the log-likelihood ratio of the same-speaker and different-speaker trial using

$$\begin{aligned} s(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) &= \boldsymbol{\phi}_1^T\mathbf{\Lambda}\boldsymbol{\phi}_2 + \boldsymbol{\phi}_2^T\mathbf{\Lambda}\boldsymbol{\phi}_1 + \boldsymbol{\phi}_1^T\mathbf{\Gamma}\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2^T\mathbf{\Gamma}\boldsymbol{\phi}_2 \\ &+ (\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2)^T\mathbf{c} + k, \end{aligned} \tag{9}$$

where $\mathbf{\Lambda}$, $\mathbf{\Gamma}$, $\mathbf{c}$, $k$ are derived from the parameters of PLDA as in [3].

### 2.5. i-vector length normalization

PLDA assumes that the input i-vectors are normally distributed. However, in earlier studies ([2]), it has been shown that this assumption is not met.

Length normalization [1, 8] of the i-vectors forces them to lie on a unity sphere, which brings them closer to the Gaussian distribution shell where most of the probability density mass is concentrated. The transformation is given as

$$\bar{\boldsymbol{\phi}} = \frac{\boldsymbol{\phi}}{\|\boldsymbol{\phi}\|} = \frac{\boldsymbol{\phi}}{\sqrt{\boldsymbol{\phi}'\boldsymbol{\phi}}} \tag{10}$$

## 3. Discriminative classifier

We describe how we train the i-vector extractor parameters $\boldsymbol{\theta}$ in order to discriminate between same-speaker and different-speaker trials, without having to explicitly model the distributions of i-vectors.

The set of training examples, which we continue referring to as training trials, comprises both different-speaker, and same-speaker trials. Let us use the coding scheme $t \in \{-1, 1\}$ to represent labels for the different-speaker, and same-speaker trials, respectively. Assigning each trial a log-likelihood ratio $s$ and the correct label $t$, the log probability of recognizing the trial correctly can be expressed as

$$\log p(t|\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = -\log(1 + \exp(-st)). \tag{11}$$

In the case of logistic regression, the objective function to be maximized is the log probability of correctly classifying all training examples, i.e., the sum of expressions (11) evaluated for all training trials. Equivalently, this can be expressed by minimizing the cross-entropy error function, which is a sum over all training trials

$$E(\boldsymbol{\theta}) = \sum_{n=1}^{N}\alpha_n E_{LR}(t_n s_n) + \frac{\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ML}}\|^2, \tag{12}$$

where the logistic regression loss function

$$E_{LR}(ts) = \log(1 + \exp(-ts)) \tag{13}$$

is simply the negative log probability (11) of correctly recognizing a trial. We have also added the regularization term $\frac{\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ML}}\|^2$, where $\lambda$ is a constant controlling the trade-off between the error function and the regularizer, and $\boldsymbol{\theta}_{\mathrm{ML}}$ is the original maximum-likelihood estimate of the given parameter. This kind of regularization is similar to the sum-of-squares penalty; however, it controls the distance from the original parameters rather than the parameter range itself. This way, optimizing the error function fine tunes the already good parameters.

The coefficients $\alpha_n$ allow us to weight individual trials. Specifically, we use them to assign different weights to same-speaker and different-speaker trials. This allows us to select

[1] Part of the factor computation is the evaluation of $\mathbf{T}'\mathbf{\Sigma}^{-1}\mathbf{f}$, where the decomposed $\mathbf{\Sigma}^{-1}$ can be projected to the neighboring terms, see [5] for detailed formulae.

a particular operating point, around which we want to optimize the performance of our system without relying on the proportion of same- and different-speaker trials in the training set. The advantage of using the cross-entropy objective for training is that it reflects performance of the system over a wide range of operating points (around the selected point).

### 3.1. Gradient evaluation

In order to numerically optimize the parameters $\boldsymbol{\theta}$, we want to express the gradient of the error function

$$\nabla E(\boldsymbol{\theta}) = \sum_{n=1}^{N} \alpha_n \frac{\partial E_{LR}(t_n s_n)}{\partial \boldsymbol{\theta}} + \lambda(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ML}}). \tag{14}$$

We see that the loss function $E_{LR}(t_n s_n)$ is not directly dependent on $\boldsymbol{\theta}$; therefore, the chain rule must be subsequently applied.

Let us start by deriving the loss function w.r.t. the direct parameters of $E_{LR}$

$$\frac{\partial E_{LR}}{\partial \boldsymbol{\theta}} = \frac{\partial E_{LR}}{\partial s} \frac{\partial s}{\partial \boldsymbol{\theta}} \tag{15}$$

The first r.h.s. fraction of (15) is defined as

$$\frac{\partial E_{LR}(ts)}{\partial s} = -t\sigma(-ts), \tag{16}$$

where $\sigma(\cdot)$ is the logistic function. Noting that the score $s$ is a function of a length-normalized i-vector pair

$$s = s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2),$$

we get

$$\frac{\partial s_n}{\partial \boldsymbol{\theta}} = \frac{s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2)}{\partial \bar{\boldsymbol{\phi}}_1} \frac{\partial \bar{\boldsymbol{\phi}}_1}{\partial \boldsymbol{\theta}} + \frac{s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2)}{\partial \bar{\boldsymbol{\phi}}_2} \frac{\partial \bar{\boldsymbol{\phi}}_2}{\partial \boldsymbol{\theta}} \tag{17}$$

From (9), knowing that $\boldsymbol{\Lambda}$ and $\boldsymbol{\Gamma}$ are symmetrical, we can derive

$$\frac{s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2)}{\partial \bar{\boldsymbol{\phi}}_1} = 2\boldsymbol{\phi}_2' \boldsymbol{\Lambda} + 2\boldsymbol{\phi}_1' \boldsymbol{\Gamma} + \mathbf{c} \tag{18}$$

Note that the two sides of the trial can be swapped so that an analogous equation applies when deriving w.r.t. $\boldsymbol{\phi}_2$. Again, we apply the chain rule to derive through the length normalization:

$$\frac{\partial \bar{\boldsymbol{\phi}}}{\partial \boldsymbol{\theta}} = \frac{\partial \bar{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{\theta}} \tag{19}$$

where

$$\frac{\partial \bar{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}} = \frac{1}{\|\boldsymbol{\phi}\|} \left( \mathbf{I} - (\bar{\boldsymbol{\phi}} \bar{\boldsymbol{\phi}}') \right). \tag{20}$$

At this point, it is trivial to express the cross-entropy $E$ as a function of some arbitrary set of $M$ i-vectors $\boldsymbol{\Phi} = (\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_M)$. With the given formulas for derivatives, it is also straightforward to express the gradient $\frac{\partial E(\boldsymbol{\Phi})}{\partial \boldsymbol{\Phi}}$. To derive through the i-vector extractor, we will make use of the chain rule for differentials, where the following holds:

$$dE = \sum_{ij} \frac{\partial E}{\partial \phi_{ij}} d\phi_{ij} = \sum_{k} \frac{\partial E}{\partial \theta_k} d\theta_k. \tag{21}$$

By making use of the matrix differentials, we can express $d\boldsymbol{\Phi}$ as a function of $d\boldsymbol{\theta}$. For the full i-vector extractor, the differential for $j$-th column of $d\boldsymbol{\Phi}$ is given as

$$d\boldsymbol{\phi}_j = -\mathbf{L}_j^{-1} d\mathbf{L}_j \mathbf{L}_j^{-1} \mathbf{T}' \mathbf{f}_j + \mathbf{L}_j^{-1} d\mathbf{T}' \mathbf{f}_j \tag{22}$$

$$d\mathbf{L}_j = \sum_c N_j^{(c)} \left( d\mathbf{T}^{(c)\prime} \mathbf{T}^{(c)} + \mathbf{T}^{(c)\prime} d\mathbf{T}^{(c)} \right) \tag{23}$$

In the case of the simplified i-vector extractor, the corresponding differentials w.r.t. the matrices $\mathbf{T}$, $\mathbf{G}$, and $\mathbf{V}$ are given respectively as

$$\begin{aligned} d\boldsymbol{\phi}_{\mathbf{T}j} &= \mathbf{G}\hat{\mathbf{L}}_j^{-1} \mathbf{G}' d\mathbf{T}' \mathbf{f}_j & (24) \\ d\boldsymbol{\phi}_{\mathbf{G}j} &= \left( d\mathbf{G}\hat{\mathbf{L}}_j^{-1} \mathbf{G}' + \mathbf{G}\hat{\mathbf{L}}_j^{-1} d\mathbf{G}' \right) \mathbf{T}' \mathbf{f}_j & (25) \\ d\boldsymbol{\phi}_{\mathbf{V}j} &= -\mathbf{G}\hat{\mathbf{L}}_j^{-1} \mathrm{Diag}(d\mathbf{V}\mathbf{n}_j) \hat{\mathbf{L}}_j^{-1} \mathbf{G}' \mathbf{T}' \mathbf{f}_j & (26) \end{aligned}$$

where $\hat{\mathbf{L}}$ is defined in (8). Substituting one of the $d\boldsymbol{\phi}$ from the above catalogue to (21), we can find the gradient $\frac{\partial E(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$.

In the case of the full i-vector extractor, the derivative can be expressed as

$$\begin{aligned} \frac{\partial E(\mathbf{T})}{\partial \mathbf{T}} = \sum_{j=1}^{M} - \left( \mathbf{L}_j^{-1} \frac{\partial E}{\partial \boldsymbol{\phi}_j'} \boldsymbol{\phi}_j' + \boldsymbol{\phi}_j \frac{\partial E}{\partial \boldsymbol{\phi}_j} \mathbf{L}_j^{-1} \right) \mathbf{T}' \mathbf{N}_j \\ + \mathbf{L}_j^{-1} \frac{\partial E}{\partial \boldsymbol{\phi}_j} \mathbf{f}_j, \end{aligned} \tag{27}$$

where $\mathbf{N}_j$ is a diagonal matrix, whose entries are $(N_j^{(1)}, \cdots, N_j^{(1)}, N_j^{(2)}, \cdots, N_j^{(2)}, \cdots)$, where every $N_j^{(i)}$ of $\mathbf{n}_j$ is expanded to match the feature dimensionality.

For the simplified i-vector extraction, the derivatives of the parameters are

$$\begin{aligned} \frac{\partial E(\mathbf{T})}{\partial \mathbf{T}} &= \sum_{j=1}^{M} \mathbf{f}_j \frac{\partial E}{\partial \boldsymbol{\phi}_j} \mathbf{G}\hat{\mathbf{L}}_j^{-1} \mathbf{G}' & (28) \\ \frac{\partial E(\mathbf{G})}{\partial \mathbf{G}} &= \sum_{j=1}^{M} \hat{\mathbf{L}}_j^{-1} \mathbf{G}' \left( \mathbf{T}' \mathbf{f}_j \frac{\partial E}{\partial \boldsymbol{\phi}_j} + \frac{\partial E}{\partial \boldsymbol{\phi}_j'} \mathbf{f}_j' \mathbf{T} \right) & (29) \\ \frac{\partial E(\mathbf{V})}{\partial \mathbf{V}} &= \sum_{j=1}^{M} -\mathbf{n}_j \left( \frac{\partial E}{\partial \boldsymbol{\phi}_j} \mathbf{G}' \circ \mathbf{f}_j' \mathbf{T} \mathbf{G} \hat{\mathbf{L}}_j^{-2} \right) & (30) \end{aligned}$$

where the $\circ$ stands for the Hadamard product.

## 4. Experiments

### 4.1. Test setup

The results of our experiments are reported on the female part of Condition 5 of the NIST 2010 speaker recognition evaluation (SRE) dataset [9]. The recognition accuracy is given as a set of equal error rate (EER), and the normalized detection cost function (DCF) as defined in both the NIST 2010 SRE task ($\mathrm{DCF}_{\mathrm{new}}$) and the previous SRE evaluations ($\mathrm{DCF}_{\mathrm{old}}$).

### 4.2. Feature extraction

In our experiments, we used cepstral features, extracted using a 25 ms Hamming window. 19 mel frequency cepstral coefficients together with log energy were calculated every 10 ms. This 20-dimensional feature vector was subjected to short time gaussianization [10] using a 3 s sliding window. Delta and double delta coefficients were then calculated using a five-frame window giving a 60-dimensional feature vector.

Segmentation was based on the Brno University of Technology (BUT) Hungarian phoneme recognizer and relative average energy thresholding. Also, short segments were pruned out, after which the speech segments were merged.

### 4.3. System Setup

One gender-independent UBM was represented as a diagonal covariance, 64-component GMM. It was trained using LDC releases of Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, and NIST 2004-2005 SRE.

The initial i-vector extractor $\mathbf{T}$ was trained on the female portion of the following telephone data: NIST SRE 2004, NIST SRE 2005, NIST SRE 2006, Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, Fisher English Parts 1 and 2, giving 8396 female speakers in 1463 hours of speech. The dimensionality of the i-vectors was set to 400. The initial orthogonalization matrix $\mathbf{G}$ was estimated using heteroscedastic linear discriminant analysis (HLDA), as described in [4].

As described in Section 2.5, length normalization was applied after i-vector extraction.

PLDA was trained using the same data set as the $\mathbf{T}$ matrix. Only the Fisher portion was trimmed off, reducing the amount of data by approximately 50%. The across-class covariance matrix (eigen-voices) was of rank 90, and the within-class covariance matrix (eigen-channels) was full-rank.

The training dataset for the discriminative training was identical to the dataset of PLDA. The cross-entropy function was evaluated on the complete trial set, i.e., all training samples were scored against each other, giving 378387 same-speaker trials, and over 468 million different-speaker trials.

### 4.4. Numerical optimization

The numerical optimization of the parameters was performed in matlab using the optimization and differentiation tools in the BOSARIS Toolkit [11]. It uses the trust region Newton conjugate gradient method, as described in [12, 13]. In addition to the first derivatives as given in Section 3.1, this method needs to evaluate the second order Hessian-vector product [14], which can be effectively computed via the 'complex step differentiation' [15].

Different values for the regularization coefficient $\lambda$ were tested. Good convergence and stability were observed when setting it to 0.2 for the full i-vector extractor parameters, and 0.8 for the simplified version. In the case of the simplified version, the matrices $\mathbf{G}$ and $\mathbf{T}$ were optimized subsequently. It was found, however, that even though optimizing $\mathbf{V}$ kept on decreasing the error function, it would always decrease the recognition performance on the test set. Different regularizers were also tested; however, it turned out that together with good initialization, the discriminative training works only as a "finetuner" of the initial parameters.

Table 1 shows the situation when training the full i-vector extractor. There is only a slight improvement in performance. In the case of the simplified i-vector extractor, the improvement is more apparent—see Table 2 for results. We see that the simplified system is still worse than the full one; however, discriminative training has shown its potential.

Table 1: *Comparison of ML and discriminatively trained full i-vector extractors in terms of normalized DCFs and EER*

| | $\mathrm{DCF}_{\mathrm{new}}$ | $\mathrm{DCF}_{\mathrm{old}}$ | EER |
|---|---|---|---|
| ML | 0.6678 | 0.2200 | 4.74 |
| discriminative | 0.6478 | 0.2144 | 4.41 |

Table 2: *Comparison of ML and discriminatively trained simplified i-vector extractors in terms of norm. DCFs and EER*

| | $\mathrm{DCF}_{\mathrm{new}}$ | $\mathrm{DCF}_{\mathrm{old}}$ | EER |
|---|---|---|---|
| ML | 0.7496 | 0.2710 | 6.18 |
| discriminative | 0.6691 | 0.2403 | 5.41 |

## 5. Conclusions

We have proposed a technique for discriminative training of the i-vector extractor parameters using cross-entropy as the error function. We have applied the technique both to the original i-vector extractor and to its simplified version. In both cases, the discriminative training was effective, giving higher relative improvement in the simplified case.

## 6. References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. PP, no. 99, pp. 1 –1, 2010.

[2] Patrick Kenny, "Bayesian speaker verification with heavy–tailed priors," in *Proc. of Odyssey 2010*, Brno, Czech Republic, June 2010, http://www.crim.ca/perso/patrick.kenny, keynote presentation.

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

[4] O. Glembek, P. Matějka, and L. Burget, "Simplification and optimization of i-vector extraction," in *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ, May 2011.

[5] P. Kenny, "Joint factor analysis of speaker and session variability : Theory and algorithms - technical report CRIM-06/08-13. Montreal, CRIM, 2005," 2005.

[6] P. Kenny, G. Boulianne, P. Oullet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

[7] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *11th International Conference on Computer Vision*, 2007, pp. 1–8.

[8] Daniel Garcia-Romero, "Analysis of i-vector length normalization in Gaussian-PLDA speaker recognition systems," in *Proc. of the International Conference on Spoken Language Processing (ICSLP)*, 2011.

[9] "National institute of standards and technology," http://www.nist.gov/speech/tests/spk/index.htm.

[10] J. Pelecanos and S. Sridharan, "Feature warping for robust speaker verification," in *Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop*, 2006, pp. 213–218.

[11] Niko Brümmer and Edwards de Villiers, "The BOSARIS toolkit," http://sites.google.com/site/bosaristoolkit/.

[12] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi, "Trust region Newton method for large-scale logistic regression," *Journal of Machine Learning Research*, Sept. 2008.

[13] Jorge Nocedal and Stephen J. Wright, *Numerical Optimization*, Springer, 2nd edition, 2006.

[14] Barak A. Pearlmutter, "Fast exact multiplication by the Hessian," *Neural Computation*, vol. 6, pp. 147–160, 1994.

[15] Lawrence F. Shampine, "Accurate numerical derivatives in MATLAB," *ACM Trans. Math. Softw.*, pp. –1–1, 2007.

# A NOISE ROBUST I-VECTOR EXTRACTOR USING VECTOR TAYLOR SERIES FOR SPEAKER RECOGNITION

*Yun Lei* [⋆] *Lukáš Burget* [†] *Nicolas Scheffer* [⋆]

[⋆] Speech Technology and Research Laboratory, SRI International, California, USA
{yunlei,scheffer}@speech.sri.com
[†] Brno University of Technology, Czech Republic
burget@fit.vutbr.cz

## ABSTRACT

We propose a novel approach for noise-robust speaker recognition, where the model of distortions caused by additive and convolutive noises is integrated into the i-vector extraction framework. The model is based on a vector taylor series (VTS) approximation widely successful in noise robust speech recognition. The model allows for extracting ”cleaned-up” i-vectors which can be used in a standard i-vector back end. We evaluate the proposed framework on the PRISM corpus, a NIST-SRE like corpus, where noisy conditions were created by artificially adding babble noises to clean speech segments. Results show that using VTS i-vectors present significant improvements in all noisy conditions compared to a state-of-the-art baseline speaker recognition. More importantly, the proposed framework is robust to noise, as improvements are maintained when the system is trained on clean data.

***Index Terms***— speaker recognition, Vector Taylor Series, i-vector, noisy speaker verification, noise compensation

## 1. INTRODUCTION

Recently, the speaker verification community has seen a significant increase in accuracy from the successful application of the i-vector extraction paradigm [1]. Along with a Bayesian back-end such as probabilistic linear discriminant analysis (PLDA) [2, 3, 4], it has become the state of the art in speaker verification. In this framework, each speech utterance with variable duration is projected into an i-vector – a single low-dimensional feature vector, typically of a few hundred components. More specifically, an i-vector is a point estimate of a latent variable vector representing a Gaussian mixture model (GMM) adapted to the corresponding utterance. A PLDA model is then used to compare i-vectors representing different utterances and to produce verification scores.

This work is focused on the robustness of speaker verification systems in the presence of noisy speech. With recent widespread use of speech-enabled services for consumers and growing importance of speaker recognition in security and defence, the need for noise-robust techniques is on the rise. Although current state-of-the-art speaker recognition systems achieve very high performance on clean data, there are few studies of noisy conditions. In a previous study [5], we have successfully proposed a robust strategy to compensate for degradations from noise by adopting a multi-style training approach for the PLDA backend. While significant improvements were obtained, worse performance by an order of magnitude are still observed when comparing clean to degraded conditions. In this work, we propose to tackle the problem at an earlier stage, where the i-vector extractor explicitly takes into account the potential degradations in the speech data.

Our approach is inspired by a successful acoustic modeling technique for noise robust automatic speech recognition (ASR) [6, 7], where a VTS approximation is used to model non-linear distortions in the mel-cepstral domain caused by both additive and convolutive noise. In ASR, the VTS approximation is used to synthesize acoustic model of noisy speech from a given clean speech model and from estimated noise distributions. Results observed in [7, 8, 9] show that a significant improvement can be obtained from the VTS approach in noisy environments.

In contrast to ASR, where VTS is used to synthesize noisy model, we use the approach in a somewhat opposite manner where our goal is to obtain a clean version of an i-vector. In our work, VTS is used to decompose the GMM adapted to a noisy speech segment into i) a clean GMM represented by ”clean” i-vector and ii) the distributions of the noise. One of the main benefit is that the resulting i-vector can be used in a standard PLDA backend.

It is worth to point out the similarity between our technique and joint factor analysis (JFA) [10], where the low-dimensional GMM representation is also decomposed into speaker and channel factors. However, the channel factors, which are responsible for modeling the unwanted variability (such as additive and convolutive noise), can only model linear additive effects in the GMM mean supervector domain. In contrast, our technique considers highly non-linear effects that an additive noise has on GMM all parameters (both means and covariances). Moreover, our noise compensation technique is integrated into the more modern i-vector framework, which has been shown to be superior to JFA [1].

## 2. UBM ADAPTATION USING VTS

The first step of the standard i-vector extraction is to compute the zero and first order sufficient statistics for a universal background model (UBM). In our approach, the sufficient statistics are collected from a noisy UBM synthesized for each speech segment using the VTS based distortion model from the UBM trained on clean data and from the additive and convolutive noise distributions. Such VTS noise adaptation is essentially the same as the one in noise robust ASR [7] for HMM models.

We first present the formulas for adapting the UBM to noisy

The research by authors at SRI International was funded through a development contract with Sandia National Laboratories (#DE-AC04-94AL85000). The views herein are those of the authors and do not necessarily represent the views of the funding agencies. This work was done while Lukáš Burget was at SRI International.

speech while assuming known distributions of the additive and convolutive noise. We then derive the expectation-maximization (EM) algorithm to estimate the noise distribution directly from the speech segments. More detailed discussion and derivation of the presented formulas can be found in [8].

### 2.1. UBM adaptation to noisy speech

The VTS approach is based on the knowledge of the speech feature extraction process. Here the mel-frequency cepstrum coefficient (MFCC) features are used to derive the adaptation formulas. In the cepstrum extraction process, the noisy speech $y$ can be modeled as

$$\mathbf{y} = \mathbf{x} + \mathbf{h} + g(\mathbf{n} - \mathbf{x} - \mathbf{h}), \tag{1}$$

where $\mathbf{y}$, $\mathbf{x}$, $\mathbf{h}$, $\mathbf{n}$ are the cepstrum vectors corresponding to the noisy speech, clean speech, channel, and additive noise, respectively. The non-linear function $g$ is:

$$g(\mathbf{n} - \mathbf{x} - \mathbf{h}) = \mathbf{C}\log(1 + \exp(\mathbf{C}^{\dagger}(\mathbf{n} - \mathbf{x} - \mathbf{h}))), \tag{2}$$

where $\mathbf{C}$ is the discrete cosine transform (DCT) matrix and $\mathbf{C}^{\dagger}$ is its pseudo-inverse.

Assuming simple Gaussian distributions for both additive and convolutive noise, the mean vector of the $m$-th component of the noise adapted UBM can be approximated using a VTS expansion at $(\boldsymbol{\mu}_{x_m 0}, \boldsymbol{\mu}_{n0}, \boldsymbol{\mu}_{h0})$ as

$$\begin{aligned} \boldsymbol{\mu}_{y_m} \approx\ & \boldsymbol{\mu}_{x_m 0} + \boldsymbol{\mu}_{h0} + g(\boldsymbol{\mu}_{n0} - \boldsymbol{\mu}_{x_m 0} - \boldsymbol{\mu}_{h0}) \\ & + \mathbf{G}_m(\boldsymbol{\mu}_{x_m} - \boldsymbol{\mu}_{x_m 0}) + \mathbf{G}_m(\boldsymbol{\mu}_h - \boldsymbol{\mu}_{h0}) \\ & + \mathbf{F}_m(\boldsymbol{\mu}_n - \boldsymbol{\mu}_{n0}), \end{aligned} \tag{3}$$

where $\boldsymbol{\mu}_{x_m}$ is the mean of the corresponding component in the clean UBM, $\boldsymbol{\mu}_n$ and $\boldsymbol{\mu}_h$ are the means of the additive and convolutive noise distributions, respectively. $\mathbf{G}_m$ and $\mathbf{F}_m$ are defined as:

$$\mathbf{G}_m = \mathbf{C} \cdot \operatorname{diag}\left(\frac{1}{1 + \exp(\mathbf{C}^{\dagger}(\boldsymbol{\mu}_{n0} - \boldsymbol{\mu}_{x_m 0} - \boldsymbol{\mu}_{h0}))}\right) \cdot \mathbf{C}^{\dagger} \tag{4}$$

$$\mathbf{F}_m = \mathbf{I} - \mathbf{G}_m. \tag{5}$$

To synthesize the noisy UBM, the VTS expansion is done at the point $(\boldsymbol{\mu}_{x_m 0} = \boldsymbol{\mu}_{x_m}, \boldsymbol{\mu}_{n0} = \boldsymbol{\mu}_n, \boldsymbol{\mu}_{h0} = \boldsymbol{\mu}_h)$, which reduces (3) to

$$\boldsymbol{\mu}_{y_m 0} \approx \boldsymbol{\mu}_{x_m 0} + \boldsymbol{\mu}_{h0} + g(\boldsymbol{\mu}_{n0} - \boldsymbol{\mu}_{x_m 0} - \boldsymbol{\mu}_{h0}). \tag{6}$$

The more general formula (3) is nevertheless useful for the the following derivations.

The noise-adapted covariance matrix can be approximated as

$$\boldsymbol{\Sigma}_{y_m} \approx \mathbf{G}_m \boldsymbol{\Sigma}_{x_m} \mathbf{G}_m^T + \mathbf{F}_m \boldsymbol{\Sigma}_n \mathbf{F}_m^T, \tag{7}$$

where $\boldsymbol{\Sigma}_{x_m}$ is covariance matrix of $m$-th Gaussian component from the clean UBM, $\boldsymbol{\Sigma}_n$ is the additive noise covariance matrix and $\boldsymbol{\Sigma}_h$ is set to zero since the channel is usually considered to be fixed.

In addition, the first and second order derivatives ($\Delta$ and $\Delta^2$) of the MFCC features are commonly used for speaker recognition. The means and covariances of these dynamic features can be approximated as

$$\boldsymbol{\mu}_{\Delta y_m} \approx \mathbf{G}_m \boldsymbol{\mu}_{\Delta x_m} \tag{8}$$

$$\boldsymbol{\Sigma}_{\Delta y_m} \approx \mathbf{G}_m \boldsymbol{\Sigma}_{\Delta x_m} \mathbf{G}_m^T + \mathbf{F}_m \boldsymbol{\Sigma}_{\Delta n} \mathbf{F}_m^T, \tag{9}$$

where we assume the noise to be stationary so that $\boldsymbol{\mu}_{\Delta n}$ and $\boldsymbol{\mu}_{\Delta h}$ are set to zero for simplicity.

### 2.2. Noise model estimation

For each utterance, we initialize our noise models using estimates from non-speech portions of the signal. Both additive and convolutive noise models are further updated using several EM iterations to better fit the noise adapted UBM to the noisy speech. The EM auxiliary function can be written as

$$\begin{aligned} Q =\ & \sum_i \sum_t \sum_m \gamma_{mt}^{(i)} \Big[ -\frac{1}{2} \log \left| \boldsymbol{\Sigma}_{y_m}^{(i)} \right| \\ & - \frac{1}{2} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m}^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m}^{(i)}) \Big], \end{aligned} \tag{10}$$

where $\gamma_{mt}^{(i)}$ is the posterior probability that the component $m$ from the current noise-adapted UBM generated the frame $t$ from speech segment $i$. Substituting (3) into (10) and solving for noise means by maximizing the EM auxiliary function gives us the following updates:

$$\begin{aligned} \boldsymbol{\mu}_n^{(i)} =\ & \boldsymbol{\mu}_{n0}^{(i)} + \left\{ \sum_{t,m} \gamma_{mt}^{(i)} (\mathbf{F}_m^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} \mathbf{F}_m^{(i)} \right\}^{-1} \\ \times\ & \left\{ \sum_{t,m} \gamma_{mt}^{(i)} (\mathbf{F}_m^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m 0}^{(i)}) \right\} \end{aligned} \tag{11}$$

$$\begin{aligned} \boldsymbol{\mu}_h^{(i)} =\ & \boldsymbol{\mu}_{h0}^{(i)} + \left\{ \sum_{t,m} \gamma_{mt}^{(i)} (\mathbf{G}_m^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} \mathbf{G}_m^{(i)} \right\}^{-1} \\ \times\ & \left\{ \sum_{t,m} \gamma_{mt}^{(i)} (\mathbf{G}_m^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m 0}^{(i)}) \right\}, \end{aligned} \tag{12}$$

where $\boldsymbol{\mu}_{y_m 0}^{(i)}$ is given by (6) and symbols with subscript 0 corresponds to the current estimates of the parameters. In ASR, the covariance matrix $\boldsymbol{\Sigma}_n$ is usually diagonalized for efficiency. In our work, however, all covariance matrices, including those in UBM, are full. Since there is no closed-form solution to estimate $\boldsymbol{\Sigma}_n$, we use the L-BFGS-B algorithm [11] to maximize the $Q$ function. For convenience, $\boldsymbol{\Sigma}_n$ is represented using its Cholesky decomposition to assure positive-definiteness of the covariance matrix during the optimization process:

$$\boldsymbol{\Sigma}_n^{(i)} = \mathbf{U}_n^{(i)\,T} \mathbf{U}_n^{(i)}, \tag{13}$$

where $\mathbf{U}_n^{(i)}$ is the upper triangle matrix. The gradient of the auxiliary function (10) w.r.t. $\mathbf{U}_n^{(i)}$ is:

$$\begin{aligned} \frac{\partial Q}{\partial \mathbf{U}_n^{(i)}} =\ & \frac{\partial}{\partial \mathbf{U}_n^{(i)}} \sum_t \sum_m \gamma_{mt}^{(i)} \Big[ -\frac{1}{2} \log \left| \boldsymbol{\Sigma}_{y_m}^{(i)} \right| \\ & - \frac{1}{2} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m}^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m}^{(i)}) \Big] \\ =\ & \sum_t \sum_m \gamma_{mt}^{(i)} \Big[ -\mathbf{U}_n^{(i)} (\mathbf{F}_m^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} \mathbf{F}_m^{(i)} \\ & + \mathbf{U}_n^{(i)} (\mathbf{F}_m^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m}^{(i)}) \\ & \times (\mathbf{y}_t^{(i)} - \boldsymbol{\mu}_{y_m}^{(i)})^T (\boldsymbol{\Sigma}_{y_m}^{(i)})^{-1} \mathbf{F}_m^{(i)} \Big]. \end{aligned}$$

For the dynamic features, the covariance matrices (e.g., $\boldsymbol{\Sigma}_{\Delta n}$ and $\boldsymbol{\Sigma}_{\Delta^2 n}$) can be estimated in a similar way. From these equations, we observe that the updates for the means and covariance matrices are not independent. Therefore, we alternate the means and covariance updates where the posteriors $\gamma_{mt}^{(i)}$ are recalculated.

## 3. NOISE COMPENSATED I-VECTOR EXTRACTION

We lay out the new i-vector framework that fits with the VTS compensation scheme proposed earlier. In the standard i-vector framework, (clean) speech frames $\mathbf{x}^{(i)}$ from $i$-th speech segment are assumed to be generated from a GMM:

$$\begin{aligned}\mathbf{x}^{(i)} &\sim \sum_m \pi_m N(\boldsymbol{\mu}_{x_m0} + \mathbf{T}_m\boldsymbol{\omega}^{(i)}, \boldsymbol{\Sigma}_{x_m}),\\ \boldsymbol{\omega}^{(i)} &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}),\end{aligned} \tag{14}$$

where $\mathcal{N}(\boldsymbol{\mu}_{x_m0}, \boldsymbol{\Sigma}_{x_m})$ and $\pi_m$ are UBM Gaussian components and their weights, $\mathbf{T}_m$ matrices describe a low-rank subspace (called total variability subspace) in which GMM means can be adapted to a particular speech segment and $\boldsymbol{\omega}^{(i)}$ is a segment-specific standard normal distributed latent vector. For a speech segment, the i-vector is extracted as the maximum a posteriori (MAP) point estimate of the latent vector $\boldsymbol{\omega}^{(i)}$.

The model for i-vector extraction can be now adapted to noise by substituting the clean model (14) into equations (3) and (7). We perform the VTS expansion at $(\boldsymbol{\mu}_{x_m0}, \boldsymbol{\mu}_{n0}, \boldsymbol{\mu}_{h0})$ that corresponds to the clean UBM and noise means estimated using the EM algorithm from the previous section (i.e. $\boldsymbol{\mu}_{n0}$ and $\boldsymbol{\mu}_{h0}$ are set to values obtained form updates (11) and (12), respectively). This results in the following noise-adapted model:

$$\mathbf{y}^{(i)} \sim \sum_m \pi_m N(\boldsymbol{\mu}^{(i)}_{y_m0} + \mathbf{G}^{(i)}_m \mathbf{T}_m \boldsymbol{\omega}^{(i)}, \boldsymbol{\Sigma}^{(i)}_{y_m}), \tag{15}$$

where $\mathbf{G}^{(i)}_m$, $\boldsymbol{\mu}^{(i)}_{y_m0}$ and $\boldsymbol{\Sigma}^{(i)}_{y_m}$ are given by equations (4), (6) and (7). This noise-adapted model can be used for i-vector extraction where the resulting i-vectors should be (to a large extent) independent of additive and convolutive noise. They can therefore better represent the remaining variability present in speech segments, which is likely to be informative for speaker recognition.

For the convenience, let us define the following statistics collected from a noisy speech segment using the noise-adapted UBM:

$$\begin{aligned}\mathbf{f}^{(i)}_{y_m} &= \sum_t \gamma^{(i)}_{mt} (\mathbf{G}^{(i)}_m)^{-1} (\mathbf{y}^{(i)}_t - \boldsymbol{\mu}^{(i)}_{y_m})\\ (\hat{\boldsymbol{\Sigma}}^{(i)}_{y_m})^{-1} &= (\mathbf{G}^{(i)}_m)^T (\boldsymbol{\Sigma}^{(i)}_{y_m})^{-1} \mathbf{G}^{(i)}_m.\end{aligned} \tag{16}$$

For a fixed soft frame alignment $\mathbf{y}^{(i)}_m$, it can be shown that the posterior distribution of $\boldsymbol{\omega}^{(i)}$ from equation (15) is Gaussian with mean and covariance matrix:

$$\begin{aligned}\langle\boldsymbol{\omega}^{(i)}\rangle &= \mathbf{L}^{(i)} \sum_m \mathbf{T}^T_m (\hat{\boldsymbol{\Sigma}}^{(i)}_{y_m})^{-1} \mathbf{f}^{(i)}_{y_m}\\ \mathbf{L}^{(i)} &= (I + \sum_m \gamma^{(i)}_m \mathbf{T}^T_m (\hat{\boldsymbol{\Sigma}}^{(i)}_{y_m})^{-1} \mathbf{T}_m)^{-1}.\end{aligned} \tag{17}$$

The i-vector extracted for segment $s$ is given by taking a MAP estimate for this distribution $\langle\boldsymbol{\omega}^{(i)}\rangle$.

Finally, we derive the corresponding EM algorithm to train the subspace parameters $\mathbf{T}_m$ in the i-vector extraction model (15). In the E-step, the posterior distribution of the latent vector $\boldsymbol{\omega}^{(i)}$ is estimated for each training segment using eq (17). The matrices $\mathbf{T}_m$ can be updated in the M-step using:

$$\begin{aligned}\mathrm{vec}(\mathbf{T}_m) &= \left(\sum_i \left(\gamma^{(i)}_{y_m} \langle\boldsymbol{\omega}^{(i)}(\boldsymbol{\omega}^{(i)})^T\rangle\right) \otimes (\hat{\boldsymbol{\Sigma}}^{(i)}_{y_m})^{-1}\right)^{-1}\\ &\quad \times \mathrm{vec} \sum_i (\hat{\boldsymbol{\Sigma}}^{(i)}_{y_m})^{-1} \mathbf{f}^{(i)}_{y_m} \langle\boldsymbol{\omega}^{(i)}\rangle^T\\ \langle\boldsymbol{\omega}^{(i)}\boldsymbol{\omega}^{(i)^T}\rangle &= \mathbf{L}^{(i)} + \langle\boldsymbol{\omega}^{(i)}\rangle\langle\boldsymbol{\omega}^{(i)}\rangle^T,\end{aligned} \tag{18}$$

where $\otimes$ is the Kronecker product and vec is an operator which creates a column vector from a matrix by stacking its columns. The i-vector model for the dynamic features is very similar to the one for the static feature replacing the calculation of $\boldsymbol{\mu}^{(i)}_{y_m}$ in (15) with $\boldsymbol{\mu}^{(i)}_{\Delta y_m} = \mathbf{G}^{(i)}_m \boldsymbol{\mu}_{\Delta 0m}$.

## 4. EXPERIMENTAL SETUP

Our speaker recognition system frontend extracts 20 MFCC coefficients (including C0), augmented with first and second order derivatives. A 512 diagonal component UBM is trained in a gender-dependent fashion on NIST telephone data from the speaker recognition evaluation (SRE) 2004 and 2005. A i-vector extractor of dimension 400 is then trained on a larger set (NIST SRE ’04, ’05, ’06, Switchboard, and Fisher). The dimensionality of i-vectors is further reduced to 200 by LDA, followed by length normalization and PLDA.

Results are shown on a part of the PRISM set described in [5, 12], where different noisy speech samples are added to the training, enrollment, and test sets without any overlap at three different signal-to-noise ratios (SNR) (20dB, 15dB, and 8dB). System performance is reported in terms of detection cost function (DCF) on three SNRs. The detection cost function (DCF) effective prior is the one from NIST SRE 2010 [13].

The baseline system employs the above configuration and uses mean and variance normalization (MVN) on the MFCC features estimated using the speech portion of the audio file. We compare this baseline system and a system where MVN was replaced by our VTS compensation. In the case of a VTS compensated system, we first train the i-vector extractor as follows:

1. A UBM model is trained on clean data, with no artificially added noise.
2. The UBM is adapted to each speech segment using 4 iterations of EM described in section 2.2, where the covariance matrices are updated in the second iteration and the means are updated in the others.
3. This noise-adapted UBM is used to extract sufficient statistics (16) from each speech segment.
4. Using 5 EM iterations from section 3:
   (a) Estimate the posterior distribution of the latent variable using (17) for each segment.
   (b) Update matrices $\mathbf{T}_m$ using (18).

After this training process, i-vectors are extracted for each enrollment and test segments using steps 2, 3 and 4a).

## 5. RESULTS

Table 1 presents the DCF performance of the baseline (MVN) and VTS system at different SNR. Two PLDA backends were evaluated:

a *clean* backend, where the model was trained exclusively on clean data; a *multistyle* backend, where the model was trained on clean and noisy data as proposed in [5]. Results clearly show a very large gains obtained using our VTS based approach over the state-of-the-art system, especially on low SNR conditions.

Although multistyle training brought a large improvement for the MVN system, the VTS system using a clean backend still outperforms the latter in the noisy conditions. A multistyle VTS system brings an additional gain which show the complementarity of both approaches. Similar behavior was observed at the equal error rate (EER). Figure 1 shows the DET curves of all four systems at a SNR of 8dB for a more detailed performance comparison.

| | clean | | multistyle | |
|---|---|---|---|---|
| Eval. condition | MVN | VTS | MVN | VTS |
| SNR=8dB | 0.975 | 0.639 | 0.810 | 0.480 |
| SNR=15dB | 0.661 | 0.269 | 0.437 | 0.234 |
| SNR=20dB | 0.350 | 0.179 | 0.260 | 0.170 |
| Clean | 0.082 | 0.146 | 0.086 | 0.145 |

**Table 1**. *DCF performance of a state-of-the-art baseline system compared to our VTS approach where both clean and multistyle backends were used. The VTS system significantly outperforms the baseline system in low SNR conditions.*

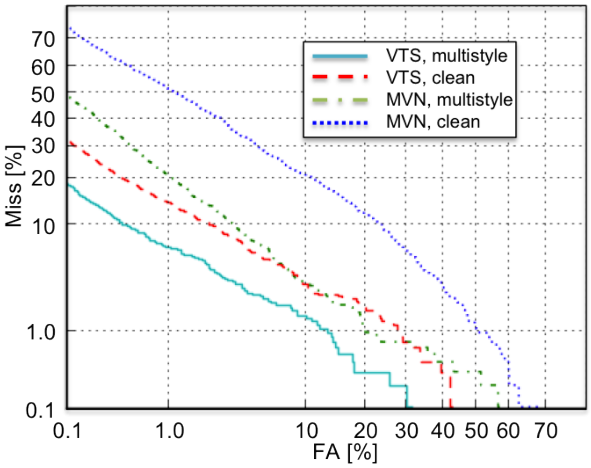


**Fig. 1**. Comparison of four systems at SNR=8dB. *MVN* means using MVN on MFCC features; *VTS* means using VTS for compensation; *clean* means using a backend model trained on clean data only; *multistyle* means using a backend model trained on clean and noisy data.

## 6. CONCLUSIONS

In this study, we successfully adapted the VTS approach to speaker recognition by proposing a new i-vector extraction framework. We show how improvements observed for VTS in speech recognition can be also obtained for speaker recognition. The proposed approach, while computationally more expensive than the standard i-vector framework, presents a relative improvement in low SNR conditions (e.g. 15 and 8db). For example, as can be also seen in figure 1, for a miss probability around 10%, the relative improvements in flase alarm rate are on the order of 70% to 80% compared to a state-of-the-art system.

We also show that our approach is robust to new and unseen data as a VTS-based system trained on clean data only outperforms a baseline system trained in a multistyle fashion in noisy conditions. This makes this approach very attractive for realistic operational scenarios where the type of degradation may not be known in advance.

We have identified two directions for future work. First, the computational requirements of the method are very high and it is impractical to scale our UBM beyond 512 Gaussians or the ivector dimension beyond 400. A substantial effort need to be put into optimizations and simplifications of the framework. Second, in speech recognition, VTS is used during the UBM model training as to 'clean up' the model for degradations caused by noise. We will explore a similar strategy for speaker recognition.

## 7. REFERENCES

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. ASLP*, vol. 19, pp. 788–798, May 2010.

[2] S.J.D. Prince, "Probabilistic linear discriminant analysis for inferences about identity," in *ICCV-11th*. IEEE, 2007, pp. 1–8.

[3] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Odyssey 2010-The Speaker and Language Recognition Workshop*. IEEE, 2010.

[4] D. Garcia-Romero and C.Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Interspeech-2011*, August 2011, pp. 249–252.

[5] Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer, "Towards noise-robust speaker recognition using probabilistic linear discriminant analysis," in *ICASSP-2012*. IEEE, March 2012, pp. 4253–4256.

[6] P. J. Moreno, *Speech recognition in noisy environments*, Ph.D. thesis, Carnegie Mellon University, 1996.

[7] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, "Hmm adaptation using vector taylor series for noisy speech recognition," in *ICSLP*, 2000, vol. 3, pp. 229–232.

[8] O. Kalinli, M. Seltzer, J. Droppo, and A Acero, "Noise adaptive training for robust automatic speech recognition," *IEEE Trans. ASLP*, vol. 18, pp. 1889–1901, Nov. 2010.

[9] H Liao, *Uncertainty Decoding for Noise Robust Speech Recognition*, PhD dissertation, University of Cambridge, Sept. 2007.

[10] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of inter-speaker variability in speaker verification," *IEEE Trans. ASLP*, vol. 16, pp. 980–988, July 2008.

[11] R. H. Byrd, P. Lu, and J. Noceda, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific and Statistical Computing*, vol. 16, pp. 1190–1208, Nov. 1995.

[12] L. Ferrer, H. Bratt, L. Burget, J. Cernocky, O. Glembek, M. Graciarena, A. Lawson, Y. Lei, P. Matejka, O. Plchot, and N. Scheffer, "Promoting robustness for speaker modeling in the community: the PRISM evaluation set," in *Proceedings of NIST 2011 Workshop*, 2011.

[13] "NIST SRE10 evaluation plan,"
http://www.itl.nist.gov/iad/mig/tests/sre/2010/NIST_SRE10_evalplan.r6.pdf.

# UNSCENTED TRANSFORM FOR IVECTOR-BASED NOISY SPEAKER RECOGNITION

*David Martínez*[1], *Lukáš Burget*[2], *Themos Stafylakis*[3], *Yun Lei*[4], *Patrick Kenny*[3], *Eduardo Lleida*[1]

[1]Aragon Institute for Engineering Research (I3A), University of Zaragoza, Spain
[2]Speech@FIT, Brno University of Technology, Czech Republic
[3]Centre de Recherche Informatique de Montreal (CRIM), Canada
[4]Speech Technology and Research Laboratory, SRI International, Menlo Park, CA, USA

## ABSTRACT

Recently, a new version of the iVector modelling has been proposed for noise robust speaker recognition, where the nonlinear function that relates clean and noisy cepstral coefficients is approximated by a first order vector Taylor series (VTS). In this paper, it is proposed to substitute the first order VTS by an unscented transform, where unlike VTS, the nonlinear function is not applied over the clean model parameters directly, but over a set of sampled points. The resulting points in the transformed space are then used to calculate the model parameters. For very low signal-to-noise ratio improvements in equal error rate of about 7% for a *clean* backend and of 14.50% for a *multistyle* backend are obtained.

***Index Terms***— Noise Robust Speaker Recognition, Unscented Transform, Vector Taylor Series, iVector

## 1. INTRODUCTION

Speaker recognition is one of the most important research fields in the speech technology industry. The main applications are found in banking, defense, forensics, video games, and also as front-end of other speech-related tasks like speech recognition. During the last decade, important technological advances have been achieved in this field. One important milestone was the development of the joint factor analysis (JFA) algorithm, a technique that makes possible to model simultaneously the inter- and intra-speaker variabilities of the features [1]. Currently, a new dimensionality reduction technique inspired by JFA is used, which allows representing a speech utterance by a low-dimension fixed length vector, or iVector, which is used for recognition [2]. The state-of-the-art recognizer is called probabilistic linear discriminant analysis (PLDA), and also allows modelling inter- and intra-speaker variability in the iVectors [3].

All these advances have brought a substantial improvement in performance and the researchers start to focus on other challenges. One important research direction is speaker recognition in noisy environments. This is not a new topic in speaker recognition [4, 5], but the interest currently lies in making the high-accuracy state-of-the-art JFA-based techniques robust to noise.

In [6], the authors present the PRISM evaluation set, a database to experiment speaker recognition systems under several noisy conditions with the aim of providing a common testbed to the community. They include language, channel, speech style, and vocal effort variabilities, also seen in NIST SRE evaluations, and other types not available on standard databases, like severe noise, and reverberation. In [7], a subset of this database is tested on different signal-to-noise ratios (SNR) and it is shown how the performance of a PLDA system modelling iVectors extracted from Mel-frequency cepstral coefficients (MFCC) is quickly degraded when the SNR decreases. It is observed that adding noisy data to the PLDA training gives relative improvements of up to 30% compared to the case where only clean data are used. The same behaviour is observed with prosodic features. By adding noisy data to train the iVector extractor no significant gains are obtained.

In [8], the authors propose a first order vector Taylor series (VTS) approximation [9] to extract noise-compensated iVectors. The approach is inspired by the VTS successfully applied in the field of automatic speech recognition (ASR) to compensate the models distorted by the nonlinear effects of noise in the cepstral domain [10, 11]. For the same PRISM subset as above, relative improvements of up to 80% compared to a state-of-the-art system with cepstral mean and variance normalization (CMVN) are observed for the speaker recognition problem, however the training process is very slow. To make it lighter, in [12] a simplified VTS (sVTS) version is proposed, where most of the improvement is kept, while the computational load is largely reduced.

In this work, the unscented transform (UT) is presented as a new approach to approximating the nonlinearity caused by noise in the cepstral domain in order to adapt the model parameters to noise. We compare UT to the first order VTS approximation. UT is a method to propagate the mean and covariance information through nonlinear transformations [13]. It is more accurate, easier to implement, and in the same order of computational expense as the linearization used with VTS, and it has been already proven to be useful for noise robust ASR [14, 15]. As shown in the experimental part of the work, UT is especially useful for very low SNR, when the nonlinear distortion is stronger.

The rest of the paper is organized as follows: in section 2 a description of the iVector approach in noisy environments is given, together with the role of VTS and UT to approximate the nonlinear relationship between clean and noisy MFCC; in section 3 the experimental part of the work is shown; and in section 4 the conclusions are drawn.

## 2. UNSCENTED TRANSFORM AND VTS IN AN IVECTOR-BASED SYSTEM

### 2.1. Standard iVector System

In the standard iVector extraction process, it is assumed that the input features, in our case MFCCs, follow a Gaussian mixture model (GMM) distribution in which the mean vector of each Gaussian is assumed to be utterance-specific. Thus the MFCCs of utterance $i$, $\mathbf{x}^{(i)}$, are evenually modelled as

$$\mathbf{x}^{(i)} \sim \sum_k \pi_k \mathcal{N}(\mu_{x_k} + \mathbf{T}_k \omega^{(i)}, \mathbf{\Sigma}_{x_k}), \tag{1}$$

being $\pi_k$, $\mu_{x_k}$, and $\mathbf{\Sigma}_{x_k}$, the weight, mean, and covariance, respectively, of Gaussian *k* of a pre-trained GMM, the universal background model (UBM), $\mathbf{T}_k$ a low-rank matrix spanning a subspace referred to as total variability subspace that describes intersession variability in the space of GMM mean parameters, and $\omega^{(i)}$ a segment-specific low-dimension latent variable with standard normal distributed prior.

The training of this model is performed via maximum likelihood (ML) in two parts. Firstly, the UBM is pre-trained using the expectation-maximization (EM) algorithm, and $\pi_k$, $\mu_{x_k}$, and $\mathbf{\Sigma}_{x_k}$ are obtained for all the Gaussians. Secondly, the sufficient statistics are computed as defined in [2] using fixed Gaussian alignments given by the UBM, and they are used for the training of the $\mathbf{T}_k$ matrices, which is also performed with the EM algorithm [2].

The iVector of utterance *i* is defined as the maximum a posteriori (MAP) point estimate of $\omega^{(i)}$. The posterior probability distribution of $\omega^{(i)}$ is Gaussian with mean, $\langle\omega^{(i)}\rangle$, and covariance, $\mathbf{L}^{(i)}$, and thus the iVector is equal to $\langle\omega^{(i)}\rangle$. The expressions to compute it are

$$\langle\omega^{(i)}\rangle = \mathbf{L}^{(i)} \sum_k \tilde{\mathbf{T}}_k^T \tilde{\mathbf{f}}_k^{(i)} \tag{2}$$

$$\mathbf{L}^{(i)} = (I + \sum_k N_{xk}^{(i)} \tilde{\mathbf{T}}_k^T \tilde{\mathbf{T}}_k)^{-1} \tag{3}$$

where $\mathbf{\Sigma}_{xk} = \mathbf{P}_{xk}\mathbf{P}_{xk}^T$, with $\mathbf{P}_{xk}$ lower triangular by Cholesky decomposition, $\tilde{\mathbf{T}}_k = \mathbf{P}_{xk}^{-1}\tilde{\mathbf{T}}_k$, and

$$N_{xk}^{(i)} = \sum_t \gamma_{xt}^{(i)}(k), \qquad \tilde{\mathbf{f}}_k^{(i)} = \mathbf{P}_{xk}^{-1} \sum_t \gamma_{xt}^{(i)}(k)(\mathbf{x}_t^{(i)} - \mu_{xk}) \tag{4}$$

are the zeroth and *whitened* first order sufficient statistics pre-collected using the UBM as proposed in [16]. The first order statistic *whitening* ($\mu_{xk}^{(i)}$ subtraction and multiplication by $\mathbf{P}_{xk}^{-1}$) not only leads to a more efficient implementation, but it also plays an important role in the sVTS approach described in section 2.3.

### 2.2. VTS-Based iVector System for Noisy Environments

According to the model of the environment presented in [9], a clean MFCC vector affected by additive and convolutional noise is distorted as

$$\mathbf{y} = \mathbf{x} + \mathbf{h} + g(\mathbf{n} - \mathbf{x} - \mathbf{h}), \tag{5}$$

where $\mathbf{y}$, $\mathbf{x}$, $\mathbf{h}$, and $\mathbf{n}$ are the cepstral vectors of the noisy speech, clean speech, channel, and additive noise, respectively, and *g* is the nonlinear function defined as

$$g = \mathbf{C}\ln(1 + exp(\mathbf{C}^{\dagger}(\mathbf{n} - \mathbf{x} - \mathbf{h}))), \tag{6}$$

with $\mathbf{C}$ and $\mathbf{C}^{\dagger}$ the discrete cosine transform matrix and its pseudo-inverse, respectively. The corresponding relationship in the model space for the UBM means [11], assuming that both types of noise follow a Gaussian distribution, is approximated by a first order VTS expansion at ($\mu_{\mathbf{x_k0}}$, $\mu_{\mathbf{h0}}$, $\mu_{\mathbf{n0}}$),

$$\begin{aligned}\mu_{y_k}^{(i)} \approx\ & \mu_{x_k0} + \mu_{h0}^{(i)} + g(\mu_{n0}^{(i)} - \mu_{x_k0} - \mu_{h0}^{(i)}) \\ & + \mathbf{G}_k^{(i)}(\mu_{x_k} - \mu_{x_k0}) + \mathbf{G}_k^{(i)}(\mu_h^{(i)} - \mu_{h0}^{(i)}) \\ & + \mathbf{F}_k^{(i)}(\mu_n^{(i)} - \mu_{n0}^{(i)}),\end{aligned} \tag{7}$$

where $\mathbf{G}_k$ is the Jacobian of *g* with respect to $\mathbf{x_k}$, and with respect to $\mathbf{h}$, and $\mathbf{F}_k$ with respect to $\mathbf{n}$. They are defined as

$$\mathbf{G}_k^{(i)} = \mathbf{C} \cdot diag(\frac{1}{1 + exp(\mathbf{C}^{\dagger}(\mu_{n0}^{(i)} - \mu_{x_k} - \mu_{h0}^{(i)}))}) \cdot \mathbf{C}^{\dagger}, \tag{8}$$

$$\mathbf{F}_k^{(i)} = \mathcal{I} - \mathbf{G}_k^{(i)}. \tag{9}$$

To compute the means of the noise-adapted UBM, $\mu_{\mathbf{y_k0}}$, the VTS is evaluated at ($\mu_{\mathbf{x_k}} = \mu_{\mathbf{x_k0}}$, $\mu_{\mathbf{h}} = \mu_{\mathbf{h0}}$, $\mu_{\mathbf{n}} = \mu_{\mathbf{n0}}$),

$$\mu_{y_k0}^{(i)} \approx \mu_{x_k0} + \mu_{h0}^{(i)} + g(\mu_{n0}^{(i)} - \mu_{x_k0} - \mu_{h0}^{(i)}) \tag{10}$$

The relationship of the UBM covariances [11], following the same reasoning as for the mean, is

$$\mathbf{\Sigma}_{y_k} \approx \mathbf{G}_k^{(i)}\mathbf{\Sigma}_{x_k}\mathbf{G}_k^{(i)T} + \mathbf{F}_k^{(i)}\mathbf{\Sigma}_n^{(i)}\mathbf{F}_k^{(i)T}, \tag{11}$$

where $\mathbf{\Sigma}_n^{(i)}$ is the additive noise covariance matrix, and $\mathbf{\Sigma}_h^{(i)}$ is set to zero since the channel is considered to be fixed. Finally, the mean and covariance of the model for the noisy MFCC first derivative ($\Delta$) are calculated with the continuous-time approximation also used in [11]. That is,

$$\mu_{\Delta y_k}^{(i)} \approx \mathbf{G}_k^{(i)}\mu_{\Delta x_k}^{(i)} \tag{12}$$

$$\mathbf{\Sigma}_{\Delta y_k}^{(i)} \approx \mathbf{G}_k^{(i)}\mathbf{\Sigma}_{\Delta x_k}\mathbf{G}_k^{(i)T} + \mathbf{F}_k^{(i)}\mathbf{\Sigma}_{\Delta n}^{(i)}\mathbf{F}_k^{(i)T}, \tag{13}$$

and identically for the MFCC second derivative ($\Delta^2$), substituting $\Delta$ by $\Delta^2$.

One important role of the VTS approximation is to make the EM objective function of the noise-adapted UBM differentiable, so closed form update formulae of the model parameters are obtained. As per [8] the objective function becomes

$$\begin{aligned}Q = & \sum_i \sum_t \sum_k \gamma_{yt}^{(i)}(k)[-\frac{1}{2}\ln|\mathbf{\Sigma}_{y_k}^{(i)}| \\ & -\frac{1}{2}(\mathbf{y}_t^{(i)} - \mu_{y_k0}^{(i)})^T(\mathbf{\Sigma}_{y_k}^{(i)})^{-1}(\mathbf{y}_t^{(i)} - \mu_{y_k0}^{(i)})],\end{aligned} \tag{14}$$

In order to include the total variability subspace in the model of the noisy MFCC of every utterance, $\mathbf{y}^{(i)}$, $\mu_{x_k}$ is substituted by $\mu_{x_k0} + T_k\omega^{(i)}$ in (7), and also considering (11), it can be shown that

$$\mathbf{y}^{(i)} \sim \sum_k \pi_k \mathcal{N}(\mu_{y_k0}^{(i)} + \mathbf{G}_k^{(i)}\mathbf{T}_k\omega^{(i)}, \mathbf{\Sigma}_{y_k}^{(i)}). \tag{15}$$

This model is trained using the EM algorithm and the equations are detailed in [8].

### 2.3. Simplified VTS

The major drawback of the VTS approach presented in previous section is the computational cost of the EM training algorithm for the total variability subspace $\mathbf{T}_k$ of (15). In particular, in the *M step* the computation of the Kronecker product and large matrix inversion given in equation (18) of [8] is several orders of magnitude more computationally and memory demanding than the calculations required for training the standard model of (1). The main differences between the two techniques are that in the VTS approach the UBM mean and covariance are utterance-dependent, and that the total variability subspace is adapted to noise differently for each utterance through the term $\mathbf{G}_k^{(i)}\mathbf{T}_k$ in (15).

In [12], a new approach is proposed that largely simplifies the equations and reduces the computational cost, the sVTS. In the sVTS, first, the UBM is adapted to each file as described in section 2.2. Then, the zeroth and *whitened* first order sufficient statistics of utterance *i* are collected over its noise-adapted UBM as

$$N_{yk}^{(i)} = \sum_t \gamma_{yt}^{(i)}(k), \qquad \tilde{\mathbf{f}}_{yk}^{(i)} = \mathbf{P}_{yk}^{(i)-1} \sum_t \gamma_{yt}^{(i)}(k)(\mathbf{y}_t^{(i)} - \mu_{y_k}^{(i)}), \tag{16}$$

where $\mathbf{\Sigma}_{y_k}^{(i)} = \mathbf{P}_{yk}^{(i)}\mathbf{P}_{yk}^{(i)T}$ by Cholesky decomposition. In this way the dependence on $\mu_{y_k}^{(i)}$, $\mathbf{\Sigma}_{y_k}^{(i)}$, and $\mathbf{G}_k^{(i)}$ completely disappears from the training, and the equations, and therefore the complexity, are reduced to the ones of the standard iVector training algorithm. Also for iVector extraction equations (2) and (3) can still be used, but replacing the sufficient statistics defined in (4) with those defined in (16). This transformation of the sufficient statistics moves the noise compensation operation to the domain of the sufficient statistics, while the former VTS approach introduced in [8] is a model domain compensation technique. In spite of the complexity reduction, the experiments made in [12] show that the sVTS preserve most of the improvements obtained with the VTS-based iVector model.

### 2.4. Simplified Unscented Transform

The UT is used to substitute the first order VTS in the model parameter adaptation. The goal is to obtain more accurate estimates of $\mu_{y_k}^{(i)}$ and $\mathbf{\Sigma}_{y_k}^{(i)}$ when the linear approximation is not good enough. The first UT method explained in [14] is followed here. Given the clean and noisy mean cepstral estimates, $\mu_{x_k 0}$ and $\mu_n^{(i)}$, an augmented signal $\hat{\mathbf{s}}_k^{(i)} = [\hat{\mathbf{x}}_k^T \;\; \hat{\mathbf{n}}^{(i)T}]^T$ is built by sampling as

$$\begin{aligned}
\hat{\mathbf{s}}_{k0}^{(i)} &= [\mu_{x_k 0}^T \;\; \mu_n^{(i)T}]^T \\
\hat{\mathbf{s}}_{kj}^{(i)} &= [\mu_{x_k 0}^T + (\sqrt{2D\mathbf{\Sigma}_{x_k}})_j \;\; \mu_n^{(i)T}]^T \\
\hat{\mathbf{s}}_{k(j+D)}^{(i)} &= [\mu_{x_k 0}^T - (\sqrt{2D\mathbf{\Sigma}_{x_k}})_j \;\; \mu_n^{(i)T}]^T \\
\hat{\mathbf{s}}_{k(j+2D)}^{(i)} &= [\mu_{x_k 0}^T \;\; \mu_n^{(i)T} + (\sqrt{2D\mathbf{\Sigma}_n^{(i)}})_j]^T \\
\hat{\mathbf{s}}_{k(j+3D)}^{(i)} &= [\mu_{x_k 0}^T \;\; \mu_n^{(i)T} - (\sqrt{2D\mathbf{\Sigma}_n^{(i)}})_j]^T
\end{aligned} \tag{17}$$

where D is the feature dimension, $j = 1...D$, therefore $\hat{\mathbf{s}}_k^{(i)}$ contains 4D+1 2D-dimension sampled vectors, and $(A)_j$ denotes the j*th* column of matrix $A$. Observe that the means and covariance matrices calculated from these samples match the actual means and covariances from which the samples were derived. Next, the sampled points are transformed using the nonlinear function

$$(f(\hat{\mathbf{s}}_k^{(i)}))_j = (\hat{\mathbf{y}}_k^{(i)})_j = (\hat{\mathbf{x}}_k)_j + \mu_{h0}^{(i)} + g((\hat{\mathbf{n}}^{(i)})_j - (\hat{\mathbf{x}}_k)_j - \mu_{h0}^{(i)}) \tag{18}$$

to obtain the noisy version of the sampled points. The mean and covariance of the noise-adapted UBM are the mean and covariance of the 4D+1 D-dimension vectors $\hat{\mathbf{y}}_k^{(i)}$, respectively,

$$\hat{\mu}_{y_k}^{(i)} = \frac{\sum_{j=0}^{4D} (\hat{\mathbf{y}}_k^{(i)})_j}{4D+1}, \tag{19}$$

$$\hat{\mathbf{\Sigma}}_{y_k}^{(i)} = \frac{\sum_{j=0}^{4D} ((\hat{\mathbf{y}}_k^{(i)})_j - \hat{\mu}_{y_k}^{(i)})((\hat{\mathbf{y}}_k^{(i)})_j - \hat{\mu}_{y_k}^{(i)})^T}{4D+1}. \tag{20}$$

Likewise, the Jacobians $\mathbf{G}_k$ and $\mathbf{F}_k$, used in the update formulae of the noise parameters and in the continuous-time approximation of the $\Delta$ and $\Delta^2$ model parameters, also depend on the sampled points and are calculated as

$$\hat{\mathbf{G}}_k^{(i)} = \frac{\sum_{j=0}^{4D} \mathbf{C} \cdot diag(\frac{1}{1+exp(\mathbf{C}^\dagger \cdot ((\hat{\mathbf{n}}^{(i)})_j - (\hat{\mathbf{x}}_k)_j - \mu_{h0}^{(i)}))}) \cdot \mathbf{C}^\dagger}{4D+1} \tag{21}$$

$$\hat{\mathbf{F}}_k^{(i)} = \mathcal{I} - \hat{\mathbf{G}}_k^{(i)} \tag{22}$$

Once the noise-adapted UBM mean and covariance, and the Jacobians are estimated, the rest of the training is exactly the same as for the VTS case. To avoid the computational complexity of the exact noise-compensated iVector extraction presented before, the simplified version is also used with the UT. Hence, this approach is named simplified UT (sUT). Note that the augmented signal contains information only of the cepstrum and not of the derivatives. These are derived through the Jacobian $\hat{\mathbf{G}}_k^{(i)}$ as per (12) and (13).

## 3. EXPERIMENTAL PART

Our features are 20 MFCC coefficients (with C0) including first and second derivatives, extracted in 25 ms long windows every 10 ms. A diagonal UBM with 512 components is trained with data coming from NIST SRE '04, '05, '06, and '08 evaluations. The 400-dimension iVector extractor is trained with data coming from NIST SRE '04, '05, '06, '08, Fisher, and Switchboard. A simplified PLDA (sPLDA) [17] with 200-dimension speaker factors is trained with the same dataset as the iVector extractor. Previously the iVectors are centered, whitened, and length-normalized [18]. Two training methods are tested for sPLDA, the *clean*, where only clean data are used, and the *multistyle*, where noisy data of 20, 15, and 8 dB are also included. The enrollment and test data is the same subset of the PRISM dataset used in [7, 8, 12]. It includes additive noise from different scenarios at three different SNRs of 20, 15, and 8 dB. Experiments are reported in terms of equal error rate (EER) and minimum of decision cost function (minDCF) as defined in [19], only on females. The SNR in enrollment and test is always the same.

In our approach, mean updates of the noise parametes **n** and **h** are obtained in the odd iterations of the EM algorithm, while the covariance update of **n** is obtained in even iterations. The reason to do it in this way is that the covariance update depends on the mean update. We have swept over several number of iterations for noise-adapted UBM training to find optimal performance. The results are obtained for the first iteration, in which only means are updated, and then every other iteration, in order to complete full updates of means and covariance.

In tables 1 and 2, the results of four different systems are compared for the *clean* sPLDA and the *multistyle* sPLDA. They are a system without noise compensation, a system with the same iVector configuration and CMVN, an sVTS system, and an sUT system. Some interesting conclusions can be found in the results. First, the *multistyle* sPLDA gives better performance than the *clean* sPLDA, as already observed in [8, 12]. Second, both the sVTS and the sUT techniques outperform CMVN, and of course, the case without noise robustness. For sVTS, iteration 3 seems to be optimal for both the *clean* and *multistyle* sPLDA. The reader should note that in every iteration the utterance-dependent log-likelihood (LLK) function of the noise-adapted UBM is increased, but this increase in LLK does not guarantee an increase in the recognition performance. We believe that more than 3 iterations overfit the data and the updates stop being useful. On the other hand, for sUT more iterations seem to be more useful. With the *clean* sPLDA iteration 7 seems to be optimal for all SNRs. For the case with *multistyle* sPLDA, the addition of noisy data in the sPLDA training makes the training to converge faster, and the best results are obtained with 3 iterations, except for the case of 8 dBs, for which the best results are obtained in iteration 5. The sUT gives better performance than the sVTS in the noisiest case, with an SNR of 8dBs. Recall that UT is an alternative to better model nonlinear distortions in the MFCC domain caused by noise, and thus, the higher the noise level, the larger the nonlinear effect, the worse the first order VTS approximation, and the larger the benefit obtained with sUT. In terms of EER and for SNR=8 dBs, with the *clean* sPLDA a 6.89% relative improvement is obtained with sUT

|  | EER(100%) |  |  |  | minDCF10 |  |  |  |
|---|---|---|---|---|---|---|---|---|
| SNR | clean | 20 dB | 15 dB | 8 dB | clean | 20 dB | 15 dB | 8 dB |
| No Robust | 1.059 | 4.179 | 14.008 | 22.135 | 0.249 | 0.489 | 0.859 | 0.946 |
| CMVN | 0.772 | 2.143 | 3.167 | 7.750 | 0.182 | 0.317 | 0.488 | 0.717 |
| sVTS it 1 | 0.851 | 1.864 | 3.029 | 7.262 | 0.197 | 0.286 | 0.451 | 0.728 |
| sVTS it 3 | 0.912 | 1.591 | 2.607 | 6.689 | 0.172 | 0.252 | 0.409 | 0.659 |
| sVTS it 5 | 0.842 | 1.765 | 2.696 | 6.478 | 0.180 | 0.284 | 0.415 | 0.697 |
| sVTS it 7 | 0.788 | 1.809 | 2.594 | 6.357 | 0.190 | 0.298 | 0.412 | 0.693 |
| sUT it 1 | 0.811 | 2.093 | 3.343 | 8.120 | 0.191 | 0.310 | 0.455 | 0.714 |
| sUT it 3 | 0.712 | 1.956 | 3.189 | 6.805 | 0.154 | 0.323 | 0.466 | 0.699 |
| sUT it 5 | 0.971 | 1.978 | 2.899 | 6.279 | 0.182 | 0.322 | 0.444 | 0.728 |
| sUT it 7 | 0.970 | 1.877 | 2.819 | 5.919 | 0.190 | 0.304 | 0.423 | 0.682 |

**Table 1**. *Results for the clean sPLDA*

|  | EER(100%) |  |  |  | minDCF10 |  |  |  |
|---|---|---|---|---|---|---|---|---|
| SNR | clean | 20 dB | 15 dB | 8 dB | clean | 20 dB | 15 dB | 8 dB |
| No Robust | 0.802 | 1.994 | 10.296 | 11.942 | 0.216 | 0.327 | 0.791 | 0.970 |
| CMVN | 0.694 | 1.786 | 2.304 | 4.261 | 0.177 | 0.278 | 0.381 | 0.635 |
| sVTS it 1 | 0.859 | 1.521 | 2.261 | 4.459 | 0.182 | 0.245 | 0.319 | 0.583 |
| sVTS it 3 | 0.846 | 1.447 | 1.918 | 4.292 | 0.169 | 0.233 | 0.338 | 0.584 |
| sVTS it 5 | 0.794 | 1.673 | 2.104 | 4.450 | 0.179 | 0.275 | 0.388 | 0.626 |
| sVTS it 7 | 0.848 | 1.790 | 2.281 | 4.514 | 0.184 | 0.276 | 0.388 | 0.627 |
| sUT it 1 | 0.844 | 1.564 | 2.311 | 4.284 | 0.191 | 0.263 | 0.334 | 0.573 |
| sUT it 3 | 0.717 | 1.412 | 1.940 | 4.087 | 0.155 | 0.241 | 0.327 | 0.568 |
| sUT it 5 | 0.879 | 1.675 | 1.975 | 3.670 | 0.148 | 0.250 | 0.306 | 0.556 |
| sUT it 7 | 0.932 | 1.639 | 2.074 | 3.708 | 0.180 | 0.260 | 0.320 | 0.582 |

**Table 2**. *Results for the multistyle sPLDA*

over sVTS, and in the *multistyle* case the relative improvement is of 14.50%, taking in both cases the optimal iterations of each technique. As final remark, note that the sVTS results are slightly different to the ones published in [12] because the feature extraction is different, and because in this work the VAD of noisy files is computed with the noisy speech, whereas there it was computed from the clean signal.

## 4. CONCLUSIONS

In this paper, the UT is presented for a speaker recognition task as an alternative to the first order VTS to approximate the nonlinearities caused by noise in the model space. The UT samples in the clean space, transforms the sampled features with the nonlinear function that relates clean and noisy MFCCs, and obtains the mean and covariances of the noise-adapted UBM in the transformed space. Unlike first order VTS, which is a linear approximation, the UT is expected to be more accurate when the distortions are far from being locally linear. The results show improvements for very low SNRs. In terms of EER, a 6.89% relative improvement is obtained for a sPLDA trained with only clean speech, and a 14.50% for a sPLDA trained with clean and noisy speech. To avoid the high computational load of the iVector modelling in the proposed noisy environment, a simplified version is followed, where the sufficient statistics are normalized with their corresponding utterance-dependent noise-adapted UBM. Finally, it is also concluded that the noise-adapted UBM calculation converges faster in sVTS than in sUT.

## 5. ACKNOWLEDGEMENTS

This work was developed thanks to the ideas shared during the workshops Bosaris 2012 and JHU 2013. Thanks to the organizers and all the participants. Thanks to Niko Brümmer for providing optimization functions, and to Olda Plchot for organizing all the data.

David Martínez was funded by the Spanish Government and the European Union (FEDER) under project TIN2011-28169-C05-02.

Lukáš Burget was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. D10PC20015, by Technology Agency of the Czech Republic grant No. TA01011328 and by IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

## 6. REFERENCES

[1] Patrick Kenny, Gilles Boulianne, Pierre Ouellet, and Pierre Dumouchel, "Speaker and Session Variability in GMM-Based Speaker Verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 4, pp. 1448–1460, May 2007.

[2] Najim Dehak, Patrick Kenny, Reda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-End Factor Analysis for Speaker Verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[3] Simon Prince and James Elder, "Probabilistic Linear Discriminant Analysis for Inferences About Identity," in *IEEE 11th International Conference on Computer Vision*, Rio de Janeiro, Brazil, 2007, pp. 1–8.

[4] Tomoko Matsui, Tomohito Kanno, and Sadaoki Furui, "Speaker Recognition Using HMM Composition in Noisy Environments," *Computer Speech & Language*, vol. 10, no. 2, pp. 107–116, 1996.

[5] Ji Ming, Timothy Hazen, James Glass, and Douglas Reynolds, "Robust Speaker Recognition in Noisy Conditions," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1711–1723, 2007.

[6] Luciana Ferrer, Harry Bratt, Lukáš Burget, Jan Černocký, Ondrej Glembek, Martin Graciarena, Aaron Lawson, Yun Lei, Pavel Matějka, Oldich Plchot, and Nicolas Scheffer, "Promoting Robustness for Speaker Modeling in the Community: the PRISM Evaluation Set," in *NIST Workshop*, Atlanta, GE, USA, 2011.

[7] Yun Lei, Lukáš Burget, and Luciana Ferrer, "Towards Noise-Robust Speaker Recognition Using Probabilistic Linear Discriminant Analysis," in *ICASSP*, Kyoto, Japan, 2012, vol. 2, pp. 4253 – 4256.

[8] Yun Lei, Lukáš Burget, and Nicolas Scheffer, "A Noise Robust iVector Extractor Using Vector Taylor Series for Speaker Recognition," in *ICASSP*, Vancouver, BC, Canada, 2013, pp. 6788 – 6791.

[9] Pedro Moreno, *Speech Recognition in Noisy Environments*, Ph.D. thesis, Carnegie Mellon University, 1996.

[10] Alex Acero, Li Deng, Trausti Kristjansson, and Jerry Zhang, "HMM Adaptation Using Vector Taylor Series for Noisy Speech Recognition," in *ICSLP*, Beijing, China, 2000, vol. 2, pp. 869–872.

[11] Ozlem Kalinli, Michael Seltzer, Jasha Droppo, and Alex Acero, "Noise Adaptive Training for Robust Automatic Speech Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 1889–1901, 2010.

[12] Yun Lei, Mitchell Mclaren, Luciana Ferrer, and Nicolas Scheffer, "Simplified VTS-Based i-Vector Extraction in Noise-Robust Speaker Recognition," in *(submitted to) ICASSP*, Florence, Italy, 2014.

[13] Simon Julier and Jeffrey Uhlmann, "Unscented Filtering and Nonlinear Estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[14] Yu Hu and Qiang Huo, "An HMM Compensation Approach Using Unscented Transformation for Noisy Speech Recognition," in *ISCSLP*, Singapore, 2006, pp. 346–357, Springer Berlin Heidelberg.

[15] Jinyu Li, Dong Yu, Yifan Gong, and Li Deng, "Unscented Transform with Online Distortion Estimation for HMM Adaptation.," in *Interspeech*, Makuhari, Japan, 2010.

[16] Ondrej Glembek, Lukáš Burget, Pavel Matějka, Martin Karafiat, and Patrick Kenny, "Simplification and Optimization of iVector Extraction," in *ICASSP*, Prague, Czech Republic, 2011, number c, pp. 4516–4519.

[17] Jesús Villalba and Eduardo Lleida, "Handling iVectors from Different Recording Conditions Using Multi-Channel Simplified PLDA in Speaker Recognition," in *ICASSP*, Vancouver, BC, Canada, 2013, pp. 6763–6767.

[18] Daniel Garcia-Romero and Carol Espy-Wilson, "Analysis of i-Vector Length Normalization in Speaker Recognition Systems.," in *Interspeech*, Florence, Italy, 2011, pp. 249–252.

[19] Alvin Martin and Craig Greenberg, "The NIST 2010 Speaker Recognition Evaluation," in *Interspeech*, Makuhari, Japan, 2010, number September, pp. 2726–2729.

# Bibliography

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

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

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

[4] Niko Brümmer, Lukáš Burget, Jan Černocký, Ondřej Glembek, František Grézl, Martin Karafiát, David van Leeuwen, Pavel Matějka, Petr Schwarz, and Albert Strasheim. Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006. *IEEE Transactions on Audio, Speech and Language Processing*, 15(7):2072–2084, 2007.

[5] Niko Brümmer and Edwards de Villiers. The speaker partitioning problem. In *Proceedings of IEEE Odyssey 2010: The Speaker and Language Recognition Workshop*, 2010.

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

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

[8] Lukáš Burget, Niko Brummer, Douglas Reynolds, Patrick Kenny, Jason Pelecanos, Robbie Vogt, Fabio Castaldo, Najim Dehak, Reda Dehak, Ondřej Glembek, Zahi Karam, Jr. John Noecker, Young Hye Na, C. Ciprian Costin, Valiantsina Hubeika, Sachin Kajarekar, Nicolas Scheffer, and Jan Černocký. Robust speaker recognition over varying channels. Technical report, Johns Hopkins University, 2008.

[9] Lukáš Burget, Pavel Matějka, Valiantsina Hubeika, and Jan Černocký. Investigation into variants of joint factor analysis for speaker recognition. In *Proc. Interspeech 2009*, pages 1263–1266, 2009.

[10] Martin Karafiát Lukáš Burget, Pavel Matějka, Ondřej Glembek, and Jan Černocký. ivector-based discriminative adaptation for automatic speech recognition. In *Proceedings of ASRU 2011*, pages 152–157. IEEE Signal Processing Society, 2011.

[11] Steven Davis and Paul Mermelstein. 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, Jul 1980.

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

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

[14] Robert B. Dunn Douglas Reynolds, Thomas F. Quatieri. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, pages 19–41, January 2000.

[15] M.J.F. Gales. Cluster adaptive training of hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, 8:417–428, 1999.

[16] Daniel Garcia-Romero. Analysis of i-vector length normalization in Gaussian-PLDA speaker recognition systems. In *Proc. of the International Conference on Spoken Language Processing (ICSLP)*, 2011.

[17] O. Glembek. *Optimization of Gaussian Mixture Subspace Models and related scoring algorithms in speaker verification.* PhD thesis, Brno University of Technology, 2009.

[18] O. Glembek, L. Burget, N. Dehak, N. Brümmer, and P. Kenny. Comparison of scoring methods used in speaker recognition with joint factor analysis. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 4057 –4060, april 2009.

[19] O. Glembek, P. Matějka, and L. Burget. Simplification and optimization of i-vector extraction. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ, May 2011.

[20] Ondřej Glembek, Lukáš Burget, Niko Bümmer, Oldřich Plchot, and Pavel Matějka. Discriminatively trained i-vector extractor for speaker verification. In *Proceedings of Interspeech 2011*, volume 2011, pages 137–140, 2011.

[21] A. O. Hatch, S. Kajarekar, and A. Stolcke. Within-Class Covariance Normalization for SVM-based speaker recognition. In *Proc. ICSLP, Pittsburgh, USA*, pages 1471–1474, 2006.

[22] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *J. Acoust. Soc. Am.*, 87:1738–1752, 1990.

[23] V. Hubeika, L. Burget, P. Matějka, and P. Schwarz. Discriminative training and channel compensation for acoustic language recognition. In *Proc. of the International Conference on Spoken Language Processing (ICSLP)*, Brisbane, Australia, September 2008.

[24] P. Kenny. Joint factor analysis of speaker and session variability: Theory and algorithms - technical report CRIM-06/08-13. Montreal, CRIM, 2005, 2005.

[25] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel. A study of interspeaker variability in speaker verification. *IEEE Transactions on Audio, Speech and Language Processing*, 16(5):980–988, July 2008.

[26] Patrick Kenny. Bayesian speaker verification with heavy–tailed priors. In *Proc. of Odyssey 2010*, Brno, Czech Republic, June 2010. http://www.crim.ca/perso/patrick.kenny, keynote presentation.

[27] Marcel Kockmann, Luciana Ferrer, Lukáš Burget, Elisabeth Shriberg, and Jan Černocký. Recent progress in prosodic speaker verification. In *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*, pages 4556–4559. IEEE Signal Processing Society, 2011.

[28] Marcel Kockmann, Luciana Ferrer, Lukáš Burget, and Jan Černocký. ivector fusion of prosodic and cepstral features for speaker verification. In *Proceedings of Interspeech 2011*, volume 2011, pages 265–268. International Speech Communication Association, 2011.

[29] Marcel Kockmann, Lukáš, and Jan Černocký. Application of speaker and language identification state-of-the-art techniques for emotion recognition. *Speech Communication*, 53(9 - 10):1172 – 1185, 2011. Sensing Emotion and Affect - Facing Realism in Speech Processing.

[30] R Kuhn, P Nguyen, J C Junqua, L Goldwasser, N Niedzielski, S Fincke, K Field, and M Contolini. Eigenvoices for speaker adaptation. In *International Conference on Spoken Language Processing*, 1998.

[31] C.J. Leggetter and P.C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech and Language*, 9(2):171 – 185, 1995.

[32] Yun Lei, Lukáš Burget, and Nicolas Scheffer. A noise robust i-vector extractor using vector taylor series for speaker recognition. In *Proceedings of ICASSP 2013*, pages 6788–6791. IEEE Signal Processing Society, 2013.

[33] Jean luc Gauvain and Chin hui Lee. Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE Transactions on Speech and Audio Processing*, 2:291–298, 1994.

[34] David González Martínez, Lukáš Burget, Luciana Ferrer, and Nicolas Scheffer. Ivector-based prosodic system for language identification. In *Acoustics, Speech, and Signal Processing, 2012. Proceedings. (ICASSP '12). IEEE International Conference on*, pages 4861–4864. IEEE Signal Processing Society, 2012.

[35] David González Martínez, Lukáš Burget, Themos Stafylakis, Yun Lei, Patrick Kenny, and Eduardo LLeida. Unscented transform for ivector-based noisy speaker recognition. In *Proceedings of ICASSP 2014*, pages 4070–4074. IEEE Signal Processing Society, 2014.

[36] David González Martínez, Oldřich Plchot, Lukáš Burget, Ondřej Glembek, and Pavel Matějka. Language recognition in ivectors space. In *Proceedings of Interspeech 2011*, volume 2011, pages 861–864. International Speech Communication Association, 2011.

[37] NIST. The NIST year 2006 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2006`, 2006.

[38] NIST. The NIST year 2008 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2008`, 2008.

[39] NIST. The NIST year 2010 speaker recognition evaluation plan. `http://www.itl.nist.gov/iad/mig//tests/sre/2010`, 2010.

[40] NIST. The NIST speaker recognition evaluation. `http://www.itl.nist.gov/iad/mig/tests/spk/`, n.d.

[41] S. J. D. Prince and J. H. Elder. Probabilistic linear discriminant analysis for inferences about identity. In *11th International Conference on Computer Vision*, 2007.

[42] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[43] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *J. Opt. Soc. Am. A*, 4(3):519–524, Mar 1987.

[44] Mohammad Mehdi Soufifar, Lukáš, Oldřich Plchot, Sandro Cumani, and Jan Černocký. Regularized subspace n-gram model for phonotactic ivector extraction. In *Proceedings of Interspeech 2013*, pages 74–78. International Speech Communication Association, 2013.

[45] Olivier Thyes, Roland Kuhn, Patrick Nguyen, and Jean-Claude Junqua. Speaker identification and verification using eigenvoices. In *INTERSPEECH*, pages 242–245, 2000.

[46] Jesús A. Villalba and Niko Brümmer. Towards fully bayesian speaker recognition: Integrating out the between-speaker covariance. In *INTERSPEECH*, pages 505–508. ISCA, 2011.

[47] R. Vogt, B. Baker, and S. Sridharan. Modelling session variability in text-independent speaker verication. In *Proc. Eurospeech*, pages 3117–3120, Lisbon, Portugal, September 2005.

[48] S Young, G Evermann, M Gales, T Hain, D Kershaw, X A Liu, G Moore, J Odell, D Ollason, D Povey, and et al. *The HTK Book*. Cambridge University Engineering Department, 2006.

[49] Xianyu Zhao and Yuan Dong. Variational bayesian joint factor analysis models for speaker verification. *Trans. Audio, Speech and Lang. Proc.*, 20(3):1032–1042, March 2012.