FACULTY
OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

REPORT

on the doctoral thesis

PLZEŇ, 2012

Ing. Lukáš Machlica

Ing. Lukáš Machlica

Vysokodimenzionální Prostory a Modelování v úloze Rozpoznávání Řečníka

obor

Kybernetika

Autoreferát disertační práce k získání akademického titulu “Doktor”

Plzeň, 24.8.2012

Ing. Lukáš Machlica

High Dimensional Spaces and Modelling in the task of Speaker Recognition

the field

Cybernetics

Report on the doctoral thesis submitted in conformity with requirements for the degree of Doctor of Philosophy

Plzeň, August 24, 2012

Disertační práce byla vypracována v prezenčním/kombinovanfm doktorském studiu na katedě kybernetiky fakulty aplikovaných věd ZČU.

Uchazeč:	Ing. Lukáš Machlica Fakulta aplikovaných věd Katedra kybernetiky Univerzitní 22, 306 14 Plzeň
Školitel:	Doc. Dr. Ing. Vlasta Radová Fakulta aplikovaných věd Katedra kybernetiky Univerzitní 22, 306 14 Plzeň
Oponenti:	Prof. Ing. Jan Uhlíř, CSc., ČVUT Praha, FEL, katedra teorie obvodů Ing. Pavel Matějka, Ph.D., VUT Brno, FIT, Ústav počítačové grafiky a multimédií

Autoreferát byl rozeslán dne: ....

Obhajoba disertační práce se koná dne .... před komisí v oboru Kybernetika na FAV ZČU, Univerzitní 22, 306 14 Plzeň, v místnosti .... v .... hodin.

S disertační prací je možno se seznámit na studijním oddělení FAV ZČU, Univerzitní 22, UV 206.

Prof. Ing. Josef Psutka CSc.
předseda oborové rady,
“Kybernetika”

Abstract

The automatic speaker recognition made a significant progress in the last two decades. Huge speech corpora containing thousands of speakers recorded on several channels are at hand, and methods utilizing as much information as possible were developed. Nowadays state-of-the-art methods are based on Gaussian mixture models used to estimate relevant statistics from feature vectors extracted from the speech of a speaker, which are further concatenated into a high dimensional vector – supervector.

Methods concerning the extraction of high dimensional supervectors along with techniques capable to build a speaker model in such a high dimensional space are described in depth and links between these methods are found. The main emphasize is laid on the analysis of these methods and an efficient implementation in order to process huge amounts of development data to train the speaker recognition system. Also the influence of development corpora on the recognition performance is experimentally tested.

Keywords: Gaussian mixture models, support vector machine, supervector, factor analysis, dimensionality reduction, speaker recognition

Abstrakt

Během posledních dvou desetiletí bylo v úloze automatického rozpoznávání řečníka dosaženo výrazných pokroků. Byly nahrány obrovské řečové databáze obsahující tisíce řečníků mluvících na různých akustických kanálech. Zároveň byly vyvinuty metody, které se snaží z těchto dat extrahovat co nejvíce informací. Nejmodernější metody jsou založeny na modelech Gaussovských směsí. S jejich pomocí jsou z příznakových vektorů, extrahovaných z řečových dat řečníků, počítány statistiky. Tyto statistiky jsou následně zřetězeny/pospojovány do vysokorozměrných vektorů – supervektorů.

Práce se zabývá podrobným popisem metod extrakce vysokodimenzionálních supervektorů společně s technikami jejich modelování. Hlavní důraz je kladen na analýzu těchto metod, jejich propojení, a protože je při trénování systému rozpoznávání řečníka potřeba zpracovat velké množství vstupních dat, i na jejich efektivní implementaci. Experimentálně je také vyšetřen vliv dat pro trénování na kvalitu rozpoznávání.

Klíčová slova: model Gaussovských směsí, support vector machine, supervektor, faktorová analýza, redukce dimenze, rozpoznávání mluvčích

Contents

List of Abbreviations

CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
EER	Equal Error Rate
EM	Expectation Maximization
FA	Factor Analysis
FC	Fusion Coefficient
FSH	Fisher English Training Speech Part 1 and Part 2
FW	Feature Warping
GLDS	Generalized Linear Discriminant Sequence
GM	Global memory
GMM	Gaussian Mixture Model
GN	Gaussian Normalization
GPU	Graphics Processing Unit
GSV	Gaussian-mean Supervector
JFA	Joint Factor Analysis
LFCC	Linear Frequency Cepstral Coefficients
LLR	Log-Likelihood Ratio
LR	Logistic Regression
LS	Least Squares
LVCSR	Large-Vocabulary Continuous Speech Recognition system
MAB	Memory-Aligned Block
MAP	Maximum A-posteriory Probability
ML	Maximum Likelihood
MLLR	Maximum Likelihood Linear Regression
NAP	Nuisance Attribute Projection
OCK	One-Class Kernel
PCA	Principal Component Analysis

PLDA	Probabilistic Linear Discriminant Analysis
RN	Rank Normalization
SIMD	Single Instruction, Multiple Data
SLK	Supervector Linear Kernel
SM	Shared Memory
SNR	Signal-to-Noise Ratio
SR	Speaker Recognition
SRE	Speaker Recognition Evaluation
SSE	Streaming SIMD Extension
SV	SuperVector
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SWC	Switchboard Cellular Audio Part 1 and Part 2
TM	Texture Memory
UBM	Universal Background Model
VAD	Voice Activity Detector

List of Functions

diag($\boldsymbol{X}$)	transforms the matrix $\boldsymbol{X}$ to a vector taking only the diagonal of the input matrix $\boldsymbol{X}$
diagz($\boldsymbol{X}$)	zeros all the non-diagonal elements of the matrix $\boldsymbol{X}$
DIAG($\boldsymbol{x}$)	creates a diagonal matrix with entries of $\boldsymbol{x}$ on its diagonal

1 Introduction

In the first phase of an automatic Speaker Recognition (SR) *feature extraction* is carried out, varying time sequence of samples (amplitudes of recorded speech wave) is processed and feature vectors are extracted. Next, a model is estimated for each speaker. For more than a decade Gaussian Mixture Models (GMMs) dominated the task of SR [1]. GMMs play still an important role in the state-of-the-art speaker recognition systems, however they are used mainly to delimit and split up the feature space according to level of interest, and to extract data statistics related to distinct parts of the feature space. This is done via estimation of an Universal Background Model (UBM) comprising many GMM components and trained on a huge amount of development data. All the acoustic conditions in which the system will be used should be covered. Subsequently, given an UBM, statistics of extracted feature vectors of a speaker, related to distinct parts of the feature space (i.e. to individual Gaussians in the UBM), are estimated. And a supervector (SV) is formed by concatenation of these statistics in accordance with GMM components in the UBM, yielding the SV of substantially high dimension.

Simultaneously two techniques to handle the high dimensional SVs were proposed. The first is based on Support Vector Machine (SVM) as a discriminative trainer [2] discussed in Section 2.1. SVM has very good generalization properties and is well suited for the task of modelling when only a few (in the case of SVs often only one) examples/supervectors of a speaker/class are available. The concept of SVs and SVM was further extended by the Nuisance Attribute Projection (NAP) [3], which is used to suppress undesirable channel variabilities between sessions (recordings of a speaker on distinct channels) of one speaker. NAP is based on an orthogonal projection, where directions most vulnerable to environment/channel changes are projected out, see Section 2.4.

The latter technique (more precisely, a set of techniques) is based on Factor Analysis (FA). The idea is that since the dimensionality of SVs is in comparison with the number of development speakers very high, many dimensions have to be correlated with each other. Hence, the effective information on the identity of speakers has to lie in a much lower subspace. Moreover, since several sessions of one speaker are available, one could determine not only the speaker identity subspace, but also the channel/session subspace, which should be also of a much lower dimension. These principles were incorporated into a method called Joint Factor Analysis (JFA) [4], where the word *joint* refers to the fact that not only the speaker, but also the channel variabilities are treated in one JFA model. However, experiments in [5] have shown, that the channel/session subspace does still contain some substantial information concerning the identity of a speaker. Therefore, JFA was extended to the concept of i-vectors, which do not distinguish between the speaker and the channel space. They work with a *total variability space* containing simultaneously speaker and channel variabilities.

Independently of JFA a method called Probabilistic Linear Discriminant Analysis (PLDA) has been developed in the computer vision to tackle the problem of face recognition [6]. PLDA is very similar to JFA, it decomposes the feature space to speaker and channel dependent subspaces, but rather than GMM based SVs ordinary feature vectors are utilized. The difference between GMM based SVs treated in JFA/i-vectors and ordinary vectors is that distinct dimensions of GMM based SVs are weighted when estimating the subspace decomposition. Since PLDA is a generative model, it allows to compute the probability that several i-vectors originate from the same source, and thus it is well suited as a verification tool for a speaker recognition system [7].

The system examined in this thesis will use SVs, SVMs, i-vectors and PLDA models along with distinct normalization techniques.

1.1 Aim of the Thesis and the Novelties

Generally, the thesis is devoted to the problem of modelling of feature sets for classification in situations where lots of data are available, but the work will be strongly oriented toward the task of open set, text independent speaker identification.

The crucial problem when implementing a state-of-the-art SR system composed of modules such as JFA, SVM, i-vector extractor or PLDA is that huge amount of development data from a lot of speakers are required, moreover several sessions have to be available for each speaker in order to train a reliable model. Therefore three main problems are faced in this thesis: *acceleration of algorithms*, *influence of development data* on the performance of the SR system, and *connections between presented methods.* However, the goal of the work is also to *describe, explain and understand the principles of individual modelling techniques* in a wider context.

In relation to the main goals mentioned above following novel approaches are presented in this work:

1. When preparing a SR system at first UBM has to be trained from a huge amount of development data (several thousands of hours, see Section 4.1). The estimation process is based on Expectation Maximization (EM) algorithm based on maximization of the data likelihood given the model. Moreover, the data statistics from the EM algorithm are utilized also in adaptation algorithms (e.g. MAP adaptation), which are used to extract SVs. Hence, it is in great demand to have a really fast and robust implementation. For this purpose parallel technologies like supercomputers, clusters, grids, and cloud infrastructures may be utilized. We have focused on the computing power provided by the Graphics Processing Unit (GPU), which is easily available and for a reasonable price. However, in order to fully exploit the potential of the GPU computing power, the algorithm has to be parallelized in a proper way and the memory management has to be handled too. For all this reasons in Chapter 5 a highly efficient parallel implementation on a GPU is proposed leading to a *hundreds times faster* EM algorithm and statistics extraction for UBM, GMM and SV estimation.

2. Next novel approach concerns the PLDA model estimation, where each speaker has to contain several sessions. The one described in [6] is extremely slow mainly for high dimensional feature vectors containing distinct numbers of examples for each individual. In addition, the operating principle of the method stays hidden. Utilizing theorems from linear algebra concerning the inversion of matrices along with some suitable rearrangements of formulas and noticing the stand-alone summation terms a much more efficient training algorithm is proposed in Section 3.2.2 and Section 3.2.3. In addition, the background of the algorithm is clarified. The revisited algorithm is thousands time faster than a naive implementation assuming a large dataset to be processed.
3. Two kinds of methods dominate the task of speaker recognition: *based on eigenvector decomposition* such as Principal Component Analysis (PCA) and Nuisance Attribute Projection (NAP), and *based on Factor Analysis* (FA). Therefore a special section is devoted to the analysis of their similarities and dissimilarities. Formulations of both methods are converted to the problem of Least Squares (LS), and in the light of LS they are simultaneously analysed. The details are given in Section 3.3.
4. In Chapter 4 experiments on state-of-the-art SR systems are performed utilizing data from NIST Speaker Recognition Evaluations (SREs) 2008 and 2010. It is shown in what extent do additional normalization techniques help to decrease the error rates, results are analysed, and also the dimensionality reduction of SVM models is examined in Section 4.5. Since SVM is used to train a speaker model associated with a speaker dependent SV, kernels (used to map – in a linear or non-linear way – input vectors to some other favourable feature space) proposed for SVs related to UBM are tested and the results are inspected.
5. Since huge amounts of development data are available, the question is how does a system behave when the amount of development data changes. The focus is laid on the system based on i-vectors and a PLDA model. Several development subsets are created and one PLDA model is trained for each of them. Moreover, two alternatives are experimentally tested: pooling all the development data and training one PLDA model, or training one PLDA model for each development subset and fusing the verification results. Experiments in Section 4.6 are provided with analysis of the results.
6. Finally, the complementarity of implemented methods is examined in Section 4.7. Since a variety of methods was tested based on generative and discriminative modelling, subspace decomposition, dimensionality reduction, etc. the combination of these methods/systems should bring additional improvements to the recognition unless one of the techniques significantly outperforms the rest of the methods. Experiments are provided with the discussion on the differences between tested methods.

2 Classifiers, Mappings and Kernels

2.1 Support Vector Machine (SVM)

SVM is a non-parametric binary classifier, where the decision boundary between two classes is given by a linear hyperplane and the task is to find a separating hyperplane so that the margin between the classes is maximized [8]. Whenever a decision of a classification depends only on a dot product of two vectors, the dot product can be replaced by a scalar *kernel function* $K(\boldsymbol{x}_1, \boldsymbol{x}_2)$, which has to satisfy certain restrictions called Mercer's conditions. These conditions specify requirements under which the output of the kernel function can be thought of as an output of a dot product of two vectors. Thus $K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \phi(\boldsymbol{x}_1)^{\mathrm{T}}\phi(\boldsymbol{x}_2)$, where $\phi(\boldsymbol{x}_i)$ is a vector function that maps $\boldsymbol{x}_i$ to some high dimensional vector (even of infinite dimension). The SVM decision function can be written as

$$f(\boldsymbol{x}_i) = \sum_{n=1}^{L} \alpha_n y_n K(\boldsymbol{x}_n, \boldsymbol{x}_i) + q, \tag{2.1}$$

and if the kernel function is linear $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$ we get

$$f(\boldsymbol{x}_i) = \left(\sum_{n=1}^{L} \alpha_n y_n \boldsymbol{x}_n^{\mathrm{T}} \right) \boldsymbol{x}_i + q = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + q, \tag{2.2}$$

where L is the number of support vectors, which combination (not necessary linear) forms the boundary, q is an offset, $\alpha_n > 0$, L and q are learned during the training process of SVM, $y_n \in \{-1, 1\}$ are the class labels, and $\boldsymbol{x}_i$ is the vector which class pertinence has to be determined, e.g. $y_i = \operatorname{sign} f(\boldsymbol{x}_i)$. SVM is trained iteratively utilizing some optimization algorithm [9]. Note that if kernel function is linear only the normal vector $\boldsymbol{w}$ and offset q of the decision boundary have to be stored. If this is not the case all the support vectors have to be stored, and in the decision process the kernel function has to be evaluated L times for each new vector in question. An example of a SVM problem is depicted in Figure 2.1.

2.2 Gaussian Mixture Model (GMM)

GMM is a parametric generative classifier. Given as a sum of weighted Gaussians $\mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{C}_m)$ with mean $\boldsymbol{\mu}_m$ and covariance $\boldsymbol{C}_m$. Hence, given a feature vector $\boldsymbol{o}_t$ its probability in a GMM is given as

$$g(\boldsymbol{o}_t) = \sum_{m=1}^{M} \omega_m \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_m, \boldsymbol{C}_m), \tag{2.3}$$

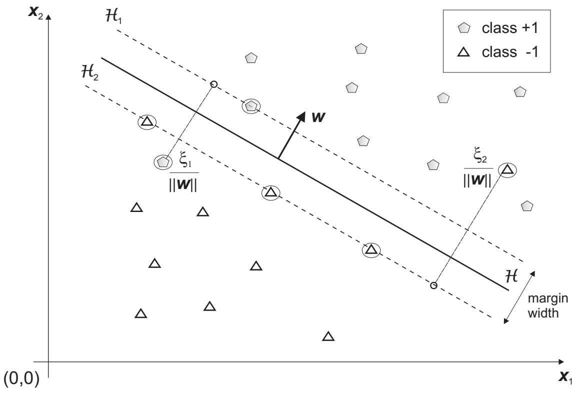

Figure 2.1: An example of the non-separable case of the SVM problem. Vectors encapsulated in circles denote support vectors.

where M is the count of Gaussians in the GMM, ω_m is the weight of m^{th} Gaussian. The most important statistic related to the m^{th} Gaussian of the GMM and a set of T_s feature vectors $\boldsymbol{O}_s = \{\boldsymbol{o}_{st}\}_{t=1}^{T_s}$ related to the s^{th} speaker is the posterior probability of m^{th} Gaussian given a feature vector $\boldsymbol{o}_{st}$:

$$\gamma_m(\boldsymbol{o}_{st}) = \frac{\omega_m \mathcal{N}(\boldsymbol{o}_{st}; \boldsymbol{\mu}_m, \boldsymbol{C}_m)}{\sum_{m=1}^{M} \omega_m \mathcal{N}(\boldsymbol{o}_{st}; \boldsymbol{\mu}_m, \boldsymbol{C}_m)}. \tag{2.4}$$

Universal Background Model (UBM) used in the task of speaker recognition is a GMM trained on a huge amount of development data containing many speakers, hence UBM is a speaker independent model describing the acoustic environment, in which the SR system is used. For further use let $D = \dim(\boldsymbol{o}_{st})$ be the dimension of feature vectors.

2.3 Mappings and Kernels

The term SuperVector (SV) used in SR is related to a high dimensional vector obtained by the concatenation of several vectors. Once a UBM was trained, supervectors

$$\boldsymbol{b}_s = \sum_{t=1}^{T_s} \left[\gamma_1(\boldsymbol{o}_{st})\boldsymbol{o}_{st}^{\mathrm{T}}, \ldots, \gamma_M(\boldsymbol{o}_{st})\boldsymbol{o}_{st}^{\mathrm{T}}\right]^{\mathrm{T}},$$

$$\boldsymbol{n}_s = \sum_{t=1}^{T_s} \left(\left[\gamma_1(\boldsymbol{o}_{st}), \ldots, \gamma_M(\boldsymbol{o}_{st})\right]^{\mathrm{T}} \otimes \boldsymbol{1}_D\right), \tag{2.5}$$

can be extracted, both are of size $DM \times 1$, $\otimes$ is the Kronecker product, $\boldsymbol{1}_D$ is a D dimensional vector of ones, and let $\boldsymbol{m}_0 = [\boldsymbol{\mu}_1^{\mathrm{T}}, \boldsymbol{\mu}_2^{\mathrm{T}}, \ldots, \boldsymbol{\mu}_M^{\mathrm{T}}]^{\mathrm{T}}$ be the SV constructed by the concatenation of the UBM means. Thus, SV in the context of this thesis is a *mapping* of a set of low dimensional feature vectors to a high dimensional representation related to a generative model.

2.3.1 GMM-mean Supervector (GSV)

GSV was proposed in [2], and it is composed of means of an Maximum A-Posteriory (MAP) adapted UBM. GSV (and also the MAP adaptation of UBM means) can be expressed as

$$\boldsymbol{\psi}^s_{\mathrm{GSV}} = \tau \boldsymbol{m}_s + (1-\tau)\boldsymbol{m}_0, \tag{2.6}$$

$$\boldsymbol{m}_s = \boldsymbol{N}_s^{-1}\boldsymbol{b}_s, \tag{2.7}$$

where $\boldsymbol{m}_s$ is the new Maximum Likelihood (ML) estimate of $\boldsymbol{m}_0$ given the dataset $\boldsymbol{O}_s$ of speaker s, $\boldsymbol{N}_s$ is a diagonal matrix with $\boldsymbol{n}_s$ on its diagonal, and τ is an empirically set parameter controlling the balance between UBM parameters $\boldsymbol{m}_0$ and the new ML estimate $\boldsymbol{m}_s$. Note that for each speaker only one SV is extracted no matter how many feature vectors are available.

Supervector Linear Kernel (SLK) It is a kernel often used in combination with GSVs and SVM [2, 10]. Assuming a UBM with diagonal covariances, the resulting kernel can be written in the form

$$K_{\mathrm{SLK}}(\boldsymbol{\psi}^s_{\mathrm{GSV}}, \boldsymbol{\psi}^q_{\mathrm{GSV}}) = \sum_{m=1}^{M} \omega_m (\boldsymbol{\mu}^s_m)^{\mathrm{T}} \boldsymbol{C}_m^{-1} \boldsymbol{\mu}^q_m = (\boldsymbol{\psi}^s_{\mathrm{GSV}})^{\mathrm{T}}\, \boldsymbol{\Omega} \boldsymbol{G}\, \boldsymbol{\psi}^q_{\mathrm{GSV}}\ , \tag{2.8}$$

$$\boldsymbol{G} : \mathrm{diag}(\boldsymbol{G}) = [\mathrm{diag}(\boldsymbol{C}_1^{-1}), \ldots, \mathrm{diag}(\boldsymbol{C}_m^{-1}), \ldots, \mathrm{diag}(\boldsymbol{C}_M^{-1})]^{\mathrm{T}}\ , \tag{2.9}$$

$$\boldsymbol{\Omega} : \mathrm{diag}(\boldsymbol{\Omega}) = [\omega_1, \ldots, \omega_M]^{\mathrm{T}} \otimes \boldsymbol{1}_D\ , \tag{2.10}$$

where $\{\omega_m, \boldsymbol{C}_m\}_{m=1}^M$ are parameters (weight and covariance of a Gaussian) of the UBM given in Section 2.2, $\boldsymbol{\mu}^s_m, \boldsymbol{\mu}^q_m$ are speaker dependent, MAP-adapted means of speaker s and speaker q, $\boldsymbol{1}_D$ is a D dimensional vector of ones, $\otimes$ denotes the Kronecker product, the function $\mathrm{diag}(\boldsymbol{X})$ transforms a matrix $\boldsymbol{X}$ to a vector with its entries equal to the diagonal of this matrix. Hence, it is a linear kernel, where each dimension of SVs is weighted according to the covariance and weight of respective Gaussian in the UBM.

2.3.2 Maximum Likelihood Linear Regression (MLLR) Supervector

The rows of adaptation matrix $\boldsymbol{W}^s = [\boldsymbol{A}^s, \boldsymbol{b}^s]$ used to adapt means of a UBM according to the data of speaker s are concatenated into a supervector $\boldsymbol{\psi}^s_{\mathrm{MLLR}}$. The MLLR adapted mean $\boldsymbol{\mu}^s_m$ of a speaker's s GMM component is given as

$$\boldsymbol{\mu}^s_m = \boldsymbol{A}^s \boldsymbol{\mu}_m + \boldsymbol{b}^s = \boldsymbol{W}^s \boldsymbol{\xi}_m, \tag{2.11}$$

where $\boldsymbol{\xi}_m = [\boldsymbol{\mu}_m^{\mathrm{T}}, 1]^{\mathrm{T}}$ and $\boldsymbol{\mu}_m$ is the mean of the UBM.

One Class Kernel (OCK) The kernel is based on the work [11]. It arises from (2.8), where means $\boldsymbol{\mu}_m$ of the UBM are transformed according to equation (2.11) yielding

$$\begin{aligned} K(\boldsymbol{\psi}^s_{\mathrm{MLLR}}, \boldsymbol{\psi}^q_{\mathrm{MLLR}}) &= \sum_{m=1}^{M} \omega_m (\boldsymbol{A}^s \boldsymbol{\mu}_m + \boldsymbol{b}^s)^{\mathrm{T}} \boldsymbol{C}_m^{-1} (\boldsymbol{A}^q \boldsymbol{\mu}_m + \boldsymbol{b}^q)\,, \\ &= (\boldsymbol{\psi}^s_{\mathrm{MLLR}})^{\mathrm{T}} \boldsymbol{Q} \boldsymbol{\psi}^q_{\mathrm{MLLR}}, \end{aligned} \tag{2.12}$$

where $[\boldsymbol{A}^s, \boldsymbol{b}^s\,]$ and $[\boldsymbol{A}^q, \boldsymbol{b}^q\,]$ are adaptation matrices for speaker s and q, respectively, ω_m, $\boldsymbol{C}_m^{-1}$ are normalization terms and parameters of the UBM, and $\boldsymbol{Q}$ is a block-diagonal matrix obtained performing some algebraic manipulations on (2.12).

2.3.3 Generalized Linear Discriminant Sequence (GLDS)

GLDS was proposed in [12]. It is based on a vector function that transforms directly the feature vectors (UBM is not involved). The SV has the form

$$\boldsymbol{\psi}^s_{\mathrm{GLDS}} = \frac{1}{T_s} \sum_{t=1}^{T_s} \boldsymbol{\varphi}(\boldsymbol{o}_{st}; k)\,, \tag{2.13}$$

where $\boldsymbol{\varphi}(\boldsymbol{o}_{st}; k)$ represents a monomial expansion of a feature vector $\boldsymbol{o}_{st}$ up to the k^{th} order, e.g. for a monomial expansion of a D dimensional feature vector $\boldsymbol{o} = [o_1, o_2, \ldots, o_D]^{\mathrm{T}}$ up to the second order we get

$$\boldsymbol{\varphi}(\boldsymbol{o}; k = 2) = [1, o_1, \ldots, o_D, o_1^2, o_1 o_2, \ldots, o_1 o_D, \tag{2.14}$$

$$o_2^2, o_2 o_3, \ldots, o_2 o_D, o_3^2, \ldots, o_D^2], \tag{2.15}$$

where $\dim(\boldsymbol{\psi}_{\mathrm{GLDS}}) = ((D + k)!)(\,D!\,k!\,)$. After substituting (2.15) into (2.13) one can notice, that the mapping (2.13) comprises first- and second-order moments – the mean and covariances of dimensions of feature vectors [13].

Covariance kernel In the case of GLDS the kernel often used is $K(\boldsymbol{\psi}^s_{\mathrm{GLDS}}, \boldsymbol{\psi}^q_{\mathrm{GLDS}}) = (\boldsymbol{\psi}^s_{\mathrm{GLDS}})^{\mathrm{T}} \boldsymbol{C}^{-1}_{\mathrm{GLDS}} \boldsymbol{\psi}^q_{\mathrm{GLDS}}$, where $\boldsymbol{C}_{\mathrm{GLDS}}$ is the covariance of GLDS SVs computed from a development set.

2.4 Nuisance Attribute Projection (NAP)

NAP is a normalization technique proposed for SVM/SV based system [3]. In cases when several recordings of a speaker are available, recorded on distinct channels (we say that several sessions of a speaker are available), the channel/session information can be utilized in order to suppress high within-speaker deviations.

The objective function minimized in NAP is given as

$$J_{\mathrm{NAP}}(\boldsymbol{P}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} w_{ij} \|\boldsymbol{P}(\boldsymbol{x}_i - \boldsymbol{x}_j)\|^2, \tag{2.16}$$

where $\boldsymbol{x}_i$ is a SV of dimension D_x, N is the number of SVs in the development set, $w_{ij} = 1$ if both $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ come from the same speaker, and 0 otherwise. $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{F}_\perp \boldsymbol{F}_\perp^{\mathrm{T}}$ is a projection matrix, $\boldsymbol{F}_\perp$ is a $D_x \times D_c$ matrix of low rank D_c, where $D_c \ll D_x$, columns of $\boldsymbol{F}_\perp$ are orthonormal, thus $\boldsymbol{F}_\perp^{\mathrm{T}} \boldsymbol{F}_\perp = \boldsymbol{I}$ and they span the subspace that is going to be projected out. It is easy to see that the properties of $\boldsymbol{P}$ are: $\boldsymbol{P}^2 = \boldsymbol{P}$ ($\boldsymbol{P}$ is idempotent) and $\boldsymbol{P} = \boldsymbol{P}^{\mathrm{T}}$ ($\boldsymbol{P}$ is symmetric). It can be shown [14] that the objective function (2.16) can be expressed as

$$J_{\mathrm{NAP}}(\boldsymbol{P}) = \mathrm{tr}(\boldsymbol{P}\boldsymbol{C}_{\mathrm{W}}) = \mathrm{tr}(\boldsymbol{C}_{\mathrm{W}}) - \mathrm{tr}(\boldsymbol{F}_\perp^{\mathrm{T}} \boldsymbol{C}_{\mathrm{W}} \boldsymbol{F}_\perp), \tag{2.17}$$

$$\boldsymbol{C}_{\mathrm{W}} = \sum_{s=1}^{S} H_s \sum_{h=1}^{H_s} (\boldsymbol{x}_{sh} - \bar{\boldsymbol{x}}_s)(\boldsymbol{x}_{sh} - \bar{\boldsymbol{x}}_s)^{\mathrm{T}}, \tag{2.18}$$

$$\bar{\boldsymbol{x}}_s = \sum_{h=1}^{H_s} \boldsymbol{x}_{sh}, \tag{2.19}$$

where H_s is the number of sessions of speaker s, S is the number of speakers in the development set, and $\boldsymbol{C}_{\mathrm{W}}$ is the weighted within-speaker covariance computed on the development set of speakers. The objective (2.17) is minimized when columns of $\boldsymbol{F}_\perp$ are formed by eigenvectors of $\boldsymbol{C}_{\mathrm{W}}$ corresponding to the D_c largest eigenvalues (highest variance is projected out).

3 Factor Analysis Based Techniques

In last years Factor Analysis (FA) based techniques gained on popularity in the task of speaker recognition when SuperVectors (SVs) were introduced. Progressive methods as Joint Factor Analysis (JFA) [4], closely related concept of i-vectors [5] or Probabilistic Linear Discriminant Analysis (PLDA) [6] are all based on FA. In JFA both within- and between-speaker subspaces are estimated *jointly* at the same time. However, since the channel component of a SV does still contain a speaker information [5] JFA was modified to the concept of i-vectors, where between both subspaces no distinction is made. All the methods are reviewed in full depth in the PhD thesis, only a short overview will be given.

3.1 Identity vectors (i-vectors)

The i-vector extractor works with SVs (2.5), hence i-vectors are related to a UBM. An assumption is met that the variation in SVs (between speakers and between sessions of a speaker) can be explained in a sufficient amount by variations of low dimensional hidden variables called identity vectors (i-vectors) [5].

The (generative) model has the form

$$\boldsymbol{\psi}_s = \boldsymbol{m}_0 + \boldsymbol{T}\boldsymbol{w}_s + \boldsymbol{\epsilon}, \quad \boldsymbol{w}_s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \;\; \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}), \tag{3.1}$$

where $\boldsymbol{\psi}_s$ is a SV of speaker s, $\boldsymbol{w}_s$ is the D_w dimensional i-vector following standard normal distribution, $\boldsymbol{T}$ is the *total variability space matrix* of size $DM \times D_w$, $\boldsymbol{m}_0$ is the mean vector of $\boldsymbol{\psi}_s$ (often mean supervector of UBM is taken instead as a good approximation), and $\boldsymbol{\epsilon}$ is a random variable describing the residual noise following normal distribution with zero mean and diagonal covariance $\boldsymbol{\Sigma}$ (its diagonal blocks are often composed from the covariances $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_m$ of the UBM).

Training

In order to train the i-vector extractor at first supervectors (2.5) are extracted for each speaker and each session of a speaker. A crucial assumption is made that each session of a speaker is in fact another speaker, hence within- and between-speaker subspace is not distinguished. Now, two steps are iterated in a sequence until predetermined number of iterations is reached:

1. for each s use previous estimate of $\boldsymbol{T}$ to extract new i-vector

$$\boldsymbol{w}_s = (\boldsymbol{I} + \boldsymbol{T}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{N}_s\boldsymbol{T})^{-1}\boldsymbol{T}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\bar{\boldsymbol{b}}_s, \tag{3.2}$$

2. let $\boldsymbol{Z} = \left(\boldsymbol{I} + \boldsymbol{T}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{N}_s\boldsymbol{T}\right)^{-1}$; use newly extracted i-vectors to compute blockwise a new estimate of $\boldsymbol{T}$

$$\boldsymbol{T}_m = \left(\sum_{s=1}^{S} \bar{\boldsymbol{b}}_{sm}\boldsymbol{w}_s^{\mathrm{T}}\right)\left(\sum_{s=1}^{S} \boldsymbol{N}_{sm}\left(\boldsymbol{w}_s\boldsymbol{w}_s^{\mathrm{T}} + \boldsymbol{Z}\right)\right)^{-1}, \tag{3.3}$$

where $\boldsymbol{N}_s$ is a diagonal matrix with $\boldsymbol{n}_s$ on its diagonal, $\bar{\boldsymbol{b}}_s = \boldsymbol{b}_s - \boldsymbol{N}_s\boldsymbol{m}_0$ is the centred version of $\boldsymbol{b}_s$ around the mean $\boldsymbol{m}_0$, and the index m in $\boldsymbol{T}_m$, $\bar{\boldsymbol{b}}_{sm}$, $\boldsymbol{n}_{sm}$ (and $\boldsymbol{N}_{sm}$) refers to blocks of $\boldsymbol{T}$, $\bar{\boldsymbol{b}}_s$, $\boldsymbol{n}_s$ (and thus to $\boldsymbol{N}_{sm}$) of sizes $D \times D_w$, $D \times 1$, $D \times 1$, respectively. Hence, $\boldsymbol{T}^{\mathrm{T}} = [\boldsymbol{T}_1^{\mathrm{T}}, \boldsymbol{T}_2^{\mathrm{T}}, \ldots, \boldsymbol{T}_{sM}^{\mathrm{T}}]$, $\bar{\boldsymbol{b}}_s^{\mathrm{T}} = [\bar{\boldsymbol{b}}_{s1}^{\mathrm{T}}, \bar{\boldsymbol{b}}_{s2}^{\mathrm{T}}, \ldots, \bar{\boldsymbol{b}}_{sM}^{\mathrm{T}}]$ and $\boldsymbol{n}_s^{\mathrm{T}} = [\boldsymbol{n}_{s1}^{\mathrm{T}}, \boldsymbol{n}_{s2}^{\mathrm{T}}, \ldots, \boldsymbol{n}_{sM}^{\mathrm{T}}]$. One can update also $\boldsymbol{\Sigma}$, for details see PhD thesis. Note that for each session of a speaker one i-vector is extracted.

In fact, the training procedure is the same as for parameters of a model of Factor Analysis (FA) differing only in the presence of $\boldsymbol{N}_s$ in estimation formulas (3.2), (3.3). If $\boldsymbol{N}_s$ would equal the identity matrix $\boldsymbol{I}$ the training procedure would be identical to the estimation procedure of parameters of a FA model, which form is identical to (3.1).

3.2 PLDA

PLDA is a generative statistical model of the form

$$\boldsymbol{x}_{ij} = \boldsymbol{\mu} + \boldsymbol{F}\boldsymbol{h}_i + \boldsymbol{G}\boldsymbol{w}_{ij} + \boldsymbol{\epsilon}_{ij} \tag{3.4}$$

where $\boldsymbol{X} = \{\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{iJ_i}\}_{i=1}^{I}$ is the set of I individuals represented as D_x dimensional vectors $\boldsymbol{x}_{ij}$, J_i is the count of distinct representations of each individual, $N = \sum_{i=1}^{I} J_i$ is the number of vectors in $\boldsymbol{X}$, and $\boldsymbol{\mu} = \mathrm{E}[\boldsymbol{x}_{ij}]$ is the mean value of vectors in $\boldsymbol{X}$. Let denote $\boldsymbol{\Lambda}_i = \{\boldsymbol{x}_{ij}\}_{j=1}^{J_i}$ the set of distinct representations of one individual. Columns of the matrix $\boldsymbol{F}$ span the between individual subspace, $\boldsymbol{h}_i$ is a D_h dimensional latent vector of coordinates in this space, it is assumed to have standard normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and represents the mutual information shared between vectors in $\boldsymbol{\Lambda}_i$. Columns of the matrix $\boldsymbol{G}$ span the within individual subspace of the space formed by vectors in $\boldsymbol{X}$, and $\boldsymbol{w}_{ij}$ is a D_w dimensional vector of coordinates in this space following standard normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. The term $\boldsymbol{\epsilon}_{ij}$ represents the residual noise factor having normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$ with diagonal covariance matrix $\boldsymbol{\Sigma}$. Thus, one can identify the identity component $\boldsymbol{\mu} + \boldsymbol{F}\boldsymbol{h}_i$ and the noise/channel component $\boldsymbol{G}\boldsymbol{w}_{ij} + \boldsymbol{\epsilon}_{ij}$ of each vector $\boldsymbol{x}_{ij}$. Note that the distribution of $\boldsymbol{x}_{ij}$ is normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{F}\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}} + \boldsymbol{\Sigma})$.

3.2.1 Training

In the training phase the parameters $\theta = \{\boldsymbol{\mu}, \boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Sigma}\}$ have to be trained. Mean $\boldsymbol{\mu}$ is estimated as the mean of all vectors $\boldsymbol{x}_{ij}$ from the development set $\boldsymbol{X}$, and to facilitate subsequent formulas let subtract $\boldsymbol{\mu}$ from all $\boldsymbol{x}_{ij}$ beforehand. In [6] a system of equations is formed

$$\begin{bmatrix} \boldsymbol{x}_{i1} \\ \vdots \\ \boldsymbol{x}_{iJ_i} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} & \boldsymbol{G} & \dots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{F} & \boldsymbol{0} & \dots & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_i \\ \boldsymbol{w}_{i1} \\ \vdots \\ \boldsymbol{w}_{iJ_i} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \\ \vdots \\ \boldsymbol{\epsilon}_{J_i} \end{bmatrix}, \hat{\boldsymbol{\Sigma}}_i = \begin{bmatrix} \boldsymbol{\Sigma} & \dots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \dots & \boldsymbol{\Sigma} \end{bmatrix}, \quad (3.5)$$

what can be written in a compact form as

$$\hat{\boldsymbol{x}}_i = \boldsymbol{A}_i \hat{\boldsymbol{y}}_i + \hat{\boldsymbol{\epsilon}}, \quad (3.6)$$

where the distribution of $\hat{\boldsymbol{\epsilon}}$ follows $\mathcal{N}(\boldsymbol{0}, \hat{\boldsymbol{\Sigma}}_i)$. Matrices $\boldsymbol{A}_i, \hat{\boldsymbol{\Sigma}}_i$ depend on i through the number of their row- and column-blocks, which are given by the number of vectors in $\boldsymbol{\Lambda}_i$. Note that the joint probability of vectors in $\boldsymbol{\Lambda}_i$ given θ equals to

$$p(\boldsymbol{\Lambda}_i|\theta) = \mathcal{N}(\hat{\boldsymbol{x}}_i|\boldsymbol{0}, \boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}} + \hat{\boldsymbol{\Sigma}}_i). \quad (3.7)$$

This formulation is equivalent to the formulation of Factor Analysis (FA) [15] and can be solved by the same estimation procedure based on maximization of (3.7). At first $\boldsymbol{h}_i, \boldsymbol{w}_{i1}, \dots, \boldsymbol{w}_{iJ_i}$ are extracted (in fact only their MAP estimates are obtained) utilizing matrices $\boldsymbol{A}_i$, $\hat{\boldsymbol{\Sigma}}_i$, hence

$$\hat{\boldsymbol{y}}_i = \left(\boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}\right)^{-1}\boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\hat{\boldsymbol{x}}_i, \quad (3.8)$$

and then $\hat{\boldsymbol{y}}_i$ is decomposed to $\boldsymbol{z}_{ij} = [\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}}]^{\mathrm{T}}, j = 1, \dots, J_i$. Finally, couples $(\boldsymbol{x}_{ij}, \boldsymbol{z}_{ij})$ are used to train $\boldsymbol{B} = [\boldsymbol{F}, \boldsymbol{G}]$ and $\boldsymbol{\Sigma}$. Update formulas are

$$\boldsymbol{Z} = \left(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{B} + \boldsymbol{I}\right)^{-1}, \quad (3.9)$$

$$\boldsymbol{B}^* = \left(\sum_{i,j}\boldsymbol{x}_{ij}\boldsymbol{z}_{ij}^{\mathrm{T}}\right)\left(\sum_{i,j}\boldsymbol{Z} + \boldsymbol{z}_{ij}\boldsymbol{z}_{ij}^{\mathrm{T}}\right)^{-1}, \quad (3.10)$$

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{ij}\operatorname{diag}\left(\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^{\mathrm{T}} - \hat{\boldsymbol{B}}\boldsymbol{z}_{ij}\boldsymbol{x}_{ij}^{\mathrm{T}}\right), \quad (3.11)$$

where $\boldsymbol{B}^*$ is the new estimate of $\boldsymbol{B}$, and the function diag() zeros the non-diagonal elements. The estimation is iterative, steps (3.8)-(3.11) have to be repeated until the convergence of (3.7) is reached.

3.2.2 Training revisited I

The problem associated with the training procedure described in the previous section is that the matrix $\boldsymbol{A}_i$ has to be reassembled, multiplied and inverted whenever the number of vectors in $\boldsymbol{\Lambda}_i$ changes in order to evaluate (3.8). Now it is going to be shown how to invert $\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}$ and adjust (3.8) leading to a much faster and easier implementation. We have to find a decomposition

$$\left(\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}\right)^{-1} = \begin{bmatrix} \boldsymbol{\Omega}_1 & \boldsymbol{\Omega}_3 \\ \boldsymbol{\Omega}_3^{\mathrm{T}} & \boldsymbol{\Omega}_2 \end{bmatrix}^{-1} = \begin{bmatrix} \tilde{\boldsymbol{\Omega}}_1 & \tilde{\boldsymbol{\Omega}}_3 \\ \tilde{\boldsymbol{\Omega}}_3^{\mathrm{T}} & \tilde{\boldsymbol{\Omega}}_2 \end{bmatrix}.$$

Choosing

$$\begin{aligned}
\boldsymbol{\Omega}_1 &= J_i \boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{F} + \boldsymbol{I}, D_h \times D_h, \\
\boldsymbol{\Omega}_2 &= \begin{bmatrix} \boldsymbol{K} & \dots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \dots & \boldsymbol{K} \end{bmatrix}, J_i D_w \times J_i D_w, \\
\boldsymbol{\Omega}_3 &= \begin{bmatrix} \boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{G} & \dots & \boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{G} \end{bmatrix}, D_h \times J_i D_w,
\end{aligned} \tag{3.12}$$

where $\boldsymbol{K} = \boldsymbol{G}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{G} + \boldsymbol{I}$. Using formulas for block inverses

$$\begin{aligned}
\tilde{\boldsymbol{\Omega}}_1 &= (\boldsymbol{\Omega}_1 - \boldsymbol{\Omega}_3 \boldsymbol{\Omega}_2^{-1} \boldsymbol{\Omega}_3^{\mathrm{T}})^{-1}, \\
\tilde{\boldsymbol{\Omega}}_3 &= -\tilde{\boldsymbol{\Omega}}_1 \boldsymbol{\Omega}_3 \boldsymbol{\Omega}_2^{-1}, \\
\tilde{\boldsymbol{\Omega}}_2 &= \boldsymbol{\Omega}_2^{-1} + \boldsymbol{\Omega}_2^{-1} \boldsymbol{\Omega}_3^{\mathrm{T}} \tilde{\boldsymbol{\Omega}}_1 \boldsymbol{\Omega}_3 \boldsymbol{\Omega}_2^{-1},
\end{aligned} \tag{3.13}$$

and after some manipulations we get formulas directly for $\boldsymbol{h}_i$, $\boldsymbol{w}_{ij}$ in the form

$$\boldsymbol{h}_i = \left(\boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{N}}^{-1} \boldsymbol{F} + 1/J_i \, \boldsymbol{I} \right)^{-1} \boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{N}}^{-1} \boldsymbol{\mu}_i, \tag{3.14}$$

$$\boldsymbol{w}_{ij} = \left(\boldsymbol{G}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{G} + \boldsymbol{I} \right)^{-1} \boldsymbol{G}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{x}_{ij} - \boldsymbol{F} \boldsymbol{h}_i \right), \tag{3.15}$$

where $\boldsymbol{\Sigma}_{\mathrm{N}} = \boldsymbol{\Sigma} + \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ and $\boldsymbol{\mu}_i = 1/J_i \sum_{j=1}^{J_i} \boldsymbol{x}_{ij}$. Hence, the latent variable $\boldsymbol{h}_i$, which represents the mutual information, is the projected mean of all the given representations of an individual i assuming full noise covariance $\boldsymbol{\Sigma}_{\mathrm{N}}$. In addition, the more representations are given the lesser the influence of the $\boldsymbol{h}_i$'s prior $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ – the term $1/J_i \boldsymbol{I}$ in (3.14). In cases where the number of representations in $\boldsymbol{\Lambda}_i$ is high the term $1/J_i \boldsymbol{I}$ in (3.14) can be left out, or fixed to an arbitrary small number to handle ill-conditioned situations. The representation dependent latent variable $\boldsymbol{w}_{ij}$ is the projection of the residual $(\boldsymbol{x}_{ij} - \boldsymbol{F}\boldsymbol{h}_i)$ on the space formed by columns of $\boldsymbol{G}$ assuming only the unexplained variance $\boldsymbol{\Sigma}$, however since only one vector is used at a time the prior is fixed. It is also obvious that $\boldsymbol{h}_i$ depends not only on one particular $\boldsymbol{x}_{ij} \in \boldsymbol{\Lambda}_i$, but on the whole set $\boldsymbol{\Lambda}_i$, whereas $\boldsymbol{w}_{ij}$ depends on $\boldsymbol{x}_{ij} \in \boldsymbol{\Lambda}_i$ and even on $\boldsymbol{h}_i$.

3.2.3 Training revisited II

The goal of the previous section was to facilitate evaluations of latent variables, now we will focus on the accumulation process in the PLDA training, more precisely on summation terms

$$\sum_{i,j} \boldsymbol{x}_{ij} \boldsymbol{z}_{ij}^{\mathrm{T}} = \sum_{i,j} \boldsymbol{x}_{ij} \left[\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}} \right], \quad \sum_{i,j} \boldsymbol{z}_{ij} \boldsymbol{z}_{ij}^{\mathrm{T}} = \sum_{i,j} \begin{bmatrix} \boldsymbol{h}_i \\ \boldsymbol{w}_{ij} \end{bmatrix} \left[\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}} \right]. \tag{3.16}$$

in (3.10) and (3.11). Substituting for $\boldsymbol{h}_i$ and for $\boldsymbol{w}_{ij}$ from (3.14) and (3.15) and rearranging the formulas, the algorithm will involve only statistics of the input data, namely $\boldsymbol{C}_X = 1/N \sum_{i,j} \boldsymbol{x}_{ij} \boldsymbol{x}_{ij}^{\mathrm{T}}$ (data covariance) and $\boldsymbol{C}_B = 1/N \sum_i J_i \boldsymbol{\mu}_i \boldsymbol{\mu}_i^{\mathrm{T}}$ (between covariance matrix). Therefore, the time to train PLDA *does not depend* on the size of the dataset. Once the matrices $\boldsymbol{C}_X$ and $\boldsymbol{C}_B$ have been estimated, the data set is no longer needed. Since the estimation process is iterative (data had to be seen/processed several times) significant computational savings may be acquired for large datasets.

3.2.4 Verification

In the verification phase two hypotheses are tested [6], namely: hypotheses $\mathcal{H}_s$ that two vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ share the same identity, and hypotheses $\mathcal{H}_d$ that the identity of two vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ differs. The Log-Likelihood Ratio (LLR) can be written as

$$\begin{aligned}\mathrm{LLR}(\boldsymbol{x}_1,\boldsymbol{x}_2) = \log\frac{p(\boldsymbol{x}_1,\boldsymbol{x}_2|\mathcal{H}_s)}{p(\boldsymbol{x}_1|\mathcal{H}_d)p(\boldsymbol{x}_2|\mathcal{H}_d)} =\\ = \log\mathcal{N}\left(\begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix};\begin{bmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}\end{bmatrix},\begin{bmatrix}\hat{\boldsymbol{C}}_X & \boldsymbol{C}_F\\ \boldsymbol{C}_F & \hat{\boldsymbol{C}}_X\end{bmatrix}\right) -\\ - \log\mathcal{N}\left(\begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix};\begin{bmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}\end{bmatrix},\begin{bmatrix}\hat{\boldsymbol{C}}_X & \boldsymbol{0}\\ \boldsymbol{0} & \hat{\boldsymbol{C}}_X\end{bmatrix}\right),\end{aligned}$$

where $\hat{\boldsymbol{C}}_X = \boldsymbol{F}\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}} + \boldsymbol{\Sigma}$ and $\boldsymbol{C}_F = \boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}$.

3.3 Relation of Factor Analysis (FA) and Nuisance Attribute Projection (NAP)

To simplify following formulas let have for each speaker s the same number of sessions N_c. Let $\boldsymbol{x}_{si}$ be a supervector (SV) extracted from each of the N recordings from the development set, and let us assume that the mean $\bar{\boldsymbol{x}}_s = 1/N_s \sum_i \boldsymbol{x}_{si}$ was already subtracted from each SV of respective speaker. Hence, the overall within covariance matrix decomposed in NAP (see Section 2.4) is $\boldsymbol{C}_{\mathrm{W}} = \sum_{s=1}^{S} N_s \sum_{i=1}^{N_s} \boldsymbol{x}_{si}\boldsymbol{x}_{si}^{\mathrm{T}} = N_c \sum_{i=1}^{N} \boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}$, thus $N_c = N_1 = \ldots = N_S$, where S is number of available speakers. To simplify the computations let us drop the scaling term N_c and let us use the overall within covariance matrix $\boldsymbol{C}_{\mathrm{W}} = 1/N \sum_{i=1}^{N} \boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}$. The NAP objective can be rewritten in the Least Square (LS) formulation as

$$\begin{aligned}J_{\mathrm{NAP}}(\boldsymbol{F}) &= \mathrm{tr}\left(\boldsymbol{P}\boldsymbol{C}_W\right) = \mathrm{tr}(\boldsymbol{C}_W) - \mathrm{tr}(\boldsymbol{C}_F)\\ &= \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \boldsymbol{F}_\perp\boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{x}_i\|^2 = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \boldsymbol{F}_\perp\boldsymbol{z}_i\|^2,\ \text{and } \boldsymbol{z}_i = \boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{x}_i,\end{aligned} \tag{3.17}$$

where $\boldsymbol{P} = I - \boldsymbol{F}_\perp\boldsymbol{F}_\perp^{\mathrm{T}}$ is the projection matrix from (2.16), columns of $\boldsymbol{F}_\perp$ are orthonormal, thus $\boldsymbol{F}_\perp\boldsymbol{z}_i$ is the orthogonal projection of $\boldsymbol{x}_i$ onto the subspace formed by columns of $\boldsymbol{F}_\perp$, and $\boldsymbol{C}_F = 1/N \sum_{i=1}^{N}(\boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{x}_i)(\boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{x}_i)^{\mathrm{T}} = 1/N\, \boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{C}_{\mathrm{W}}\boldsymbol{F}_\perp^{\mathrm{T}}$.

In the case of FA, the objective function (written in the form of LS)

$$\frac{2}{N}\,J_{\mathrm{FA}}(\boldsymbol{F}) = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \boldsymbol{F}\boldsymbol{z}_i\|^2 + \mathrm{tr}(\boldsymbol{F}\boldsymbol{H}\boldsymbol{F}^{\mathrm{T}}), \tag{3.18}$$

have to be minimized, where $\boldsymbol{z}_i = (\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{F}+\boldsymbol{I})^{-1}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_i$, and $\boldsymbol{H} = (\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{F}+\boldsymbol{I})^{-1}$. However, be aware that columns of $\boldsymbol{F}$ in NAP are assumed to be orthogonal, whereas none assumptions are made in FA.

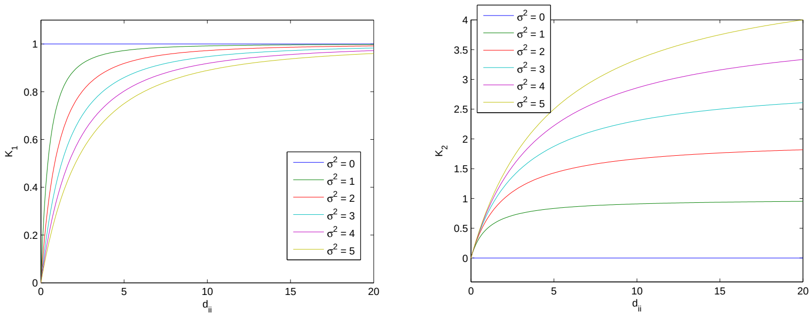

Figure 3.1: The dependency of diagonal entries of $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ on different values of d_{ii} and σ^2.

We will come out of conclusions made in [16], assuming isotropic noise $\boldsymbol{\Sigma} = \sigma^2\boldsymbol{I}$ we get $\boldsymbol{z}_i = (\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}_i$, $\boldsymbol{H} = \sigma^2(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \sigma^2\boldsymbol{I})^{-1}$. And the criterion (3.18) can be written in the form (see PhD thesis or [14])

$$\frac{2}{N} J_{\mathrm{FA}} = \mathrm{tr}(\boldsymbol{C}_W) - \mathrm{tr}(\boldsymbol{K}_1\boldsymbol{C}_F - \boldsymbol{K}_2), \tag{3.19}$$

where $\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{Q}$ was decomposed via SVD yielding $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q} = \boldsymbol{I}$, and since $\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}$ is positive semi-definite matrix, $\boldsymbol{D} = [d_{ii}]$ is a diagonal matrix with $d_{ii} \geq 0$, $\boldsymbol{F}_\perp = \boldsymbol{F}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{D}^{-1/2}$, $\boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{F}_\perp = \boldsymbol{I}$.

Thus $\boldsymbol{F}_\perp\boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{x}_i$ is an orthogonal projection onto the subspace spanned by columns of $\boldsymbol{F}$ and $\boldsymbol{F}_\perp^{\mathrm{T}}\boldsymbol{x}_i$ are the coordinates of $\boldsymbol{x}_i$ in this space,

$$\boldsymbol{K}_1 = \left[\frac{d_{ii}^2 + 2d_{ii}\sigma^2}{(d_{ii} + \sigma^2)^2}\right], \quad \boldsymbol{K}_2 = \left[\frac{d_{ii}\sigma^2}{d_{ii} + \sigma^2}\right] \tag{3.20}$$

are diagonal matrices, the dependence of their diagonal entries on d_{ii} and σ^2 is depicted in Figure 3.1, and $\mathrm{tr}(\boldsymbol{K}_2) = \mathrm{tr}(\boldsymbol{F}\boldsymbol{H}\boldsymbol{F}^{\mathrm{T}})$. Thus, $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ depend on the diagonal matrix $\boldsymbol{D}$, and $\boldsymbol{C}_F$ depends on $\boldsymbol{F}_\perp$. Note that since $\boldsymbol{K}_2$ is a diagonal matrix consisting of singular values of $\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}$ and diagonal entries of $\boldsymbol{K}_2 \geq 0$, the second term in (3.18) is responsible only for the scaling of basis vectors of the subspace formed by columns of $\boldsymbol{F}$ according to the level of noise. If $\sigma^2 >> d_{ii}$ then the corresponding directions do not contribute to minimize J_{FA} (see Figure 3.1 and the role of $\boldsymbol{K}_1$ in (3.19)), and the task of $\boldsymbol{K}_2$ is to completely eliminate these directions (diagonal elements of $\boldsymbol{K}_2$ are lower bounded by 0, see Figure 3.1).

Since $\boldsymbol{K}_1$, $\boldsymbol{K}_2$ perform only scaling of directions, in order to minimize (3.19) at first $\mathrm{tr}(\boldsymbol{C}_F)$ has to be maximized. This is done when columns of $\boldsymbol{F}_\perp$ are formed by eigenvectors of $\boldsymbol{C}_{\mathrm{W}}$ corresponding to highest eigenvalues. It can be shown (PhD thesis) that when minimizing (3.19) according to d_{ii} we get $d_{ii} = 2\lambda_i - \sigma^2$, where λ_i are the eigenvalues of $\boldsymbol{C}_{\mathrm{W}}$. Since $d_{ii} \geq 0$, a condition $d_{ii} = 0$ if $\lambda_i \leq \sigma^2/2$ has to be introduced, which is in accordance with previous discussion on the role of $\boldsymbol{K}_2$.

Comparing NAP and FA through (3.17) and (3.19) we can see that if the noise model in FA is isotropic the solutions (more precisely the estimated subspaces) for NAP and FA become identical if for all eigenvalues $\lambda_i > \sigma^2/2$. Otherwise, the FA subspace will be a also a subspace of the NAP subspace. However, criteria J_{NAP} and J_{FA} will still differ in some extent ($\boldsymbol{K}_1$ and $\boldsymbol{K}_2$). If $\sigma^2 = 0$ then $\boldsymbol{K}_1 = \boldsymbol{I}$, $\boldsymbol{K}_2 = \mathbf{0}$ and both criteria become equivalent. The same is true if we put an orthonormal restriction on columns of $\boldsymbol{F}$, hence $\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{Q}$ and $d_{ii} = 1$. Now, both $K_1 = (1 + 2\sigma^2)/(1 + \sigma^2)^2$ and $K_2 = \sigma^2/(1 + 2\sigma^2)$ become constants independent of the choice of $\boldsymbol{F}$, and $J_{\mathrm{FA}} = \alpha_1 J_{\mathrm{NAP}} + \alpha_2$ becomes a scaled version of J_{NAP} for some constants α_1, α_2. Generally, the FA criterion does incorporate also the influence of noise, thus the value of the criterion differs from J_{NAP} even if the resulting subspaces are identical.

4 Experiments

In this chapter experiments will be performed utilizing GMM-mean based, GLDS based and MLLR based supervectors (SVs). Models for SVs will be estimated mainly utilizing the Support Vector Machines (SVMs). Several distinct kernel types and their influence on the speaker recognition will be analysed along with distinct normalizations.

Further, system based on i-vectors (see Section 3.1) will be trained and a PLDA model from Section 3.2 will be used as a generative model (hence as a verification tool) in the total variability space. Since the PLDA training procedure proposed in Section 3.2.3 enables a lot of experiments to be performed, the influence of different values of latent dimensions and the influence of distinct development corpora on the PLDA estimation will be analysed.

In summary, experiments will be focused on:

1. (baseline) experiments utilizing GMM/UBM based system, where each speaker's GMM is MAP adapted and the Log-Likelihood Ratio (LLR) is computed to get a verification score
2. influence of normalization of SVs on the SVM modelling and speaker recognition
3. dimensionality reduction of SVM models
4. i-vector extraction and PLDA based generative models in the total variability space
5. influence of development speech corpora on the verification rates utilizing PLDA models
6. analysis of the complementarity of mentioned techniques

Results will be presented on two NIST Speaker Recognition Evaluation (SRE) corpora, namely NIST SRE 2008 and NIST SRE 2010, results will be given in terms of error rates – Equal Error Rate (EER) will be used, for details and other types of error rates see the full PhD thesis.

4.1 Used Corpora

In order to be able to perform reliable tests following corpora were utilized: NIST SRE 2004 (NIST04), NIST SRE 2005 (NIST05), NIST SRE 2006 (NIST06), Switchboard 1 Release 2 (SW1), Switchboard 2 Phase 3 (SW2), Switchboard Cellular Audio Part 1 and Part 2 (SWC) and Fisher English Training Speech Part 1 and Part 2 (FSH) for development purposes, and NIST SRE 2008 (NIST08), NIST SRE 2010 (NIST10) were used for calibration and/or testing purposes. The summary on corpora is given in Table 4.1 and Table 4.2, note that the development set NIST040506 contains data from NIST04, NIST05, NIST06. Each of the recordings (except those in FSH) had approximately 5 minutes in duration including the silence, length of

Table 4.1: Summary of number of recordings (REC), average number of sessions (SESS) and number of speakers (SPs) in distinct corpora.

	female			male		
corpus ID	#REC	#SESS	#SPs	#REC	#SESS	#SPs
NIST040506	5500	8	690	3787	8	465
SW1	2311	12	197	2342	11	211
SW2	2862	10	285	2183	10	216
SWC	3753	12	311	2707	12	232
FSH	5774	3	1905	4923	3	1612
overall	20200	-	3388	15942	-	2736

Table 4.2: Number of speakers and number of trials in NIST08 and NIST10.

	NIST08		NIST10	
	female	male	female	male
#target speakers	1140	648	1739	1394
#test segment speakers	2498	1535	3267	2474
#trials	28536	16968	103062	74762
#non-target trials	25157	15043	101977	73812
#target trials	3379	1925	1085	950

recordings in FSH was varying from 6 minutes to 12 minutes. The data source of all the recordings was a *telephone conversation*, however in NIST040506 also a few microphone interviews are present. In all trials only speakers of the same gender were scored against each other. Since the information concerning the gender of the speakers was known, the presented results will be given separately for each gender.

4.2 Feature Extraction

The sample rate of all the telephone speech recordings was 8000 Hz and the sampling format was μ-law. The feature extraction was based on Linear Frequency Cepstral Coefficients (LFCCs), 25 triangular filter banks were spread linearly across the frequency spectrum, 20 LFCCs were extracted, and delta coefficients were added leading to $D = 40$ dimensional feature vectors. Next, Voice Activity Detector (VAD) was used in order to discard the non-speech frames. The Feature Warping (FW – see PhD thesis) was applied after the delta coefficients were added, hence variance in each dimension was 1.

4.3 System Setup

UBM At first, two gender dependent Universal Background Models (UBMs) were estimated, one for male and one for female speakers. Each UBM (in fact a GMM) had

$M = 1024$ Gaussians and was trained on pooled development corpora NIST040506, SW1, SW2, SWC and FSH from recordings of the respective gender utilizing EM algorithm and 32 reestimations.

GMMs of Speakers GMMs of individual speakers were MAP adapted, only means were adapted with a relevance factor $\tau = 14$) using the UBM of respective gender.

Supervector (SV) Extraction Three types of SVs were extracted:

1. ψ_{GSV} – mean of each speaker's GMM were concatenated into one SV of dimension $M \times D = 1024 \times 40 = 40960$; the mapping is specified in (2.6)
2. ψ_{GLDS} – monomials up to the $k = 3^{\mathrm{rd}}$ order were expanded, and a $\frac{(D+k)!}{D!k!} = 12341$ dimensional SV was extracted; the mapping is specified in (2.13) and the monomial expansion is given in (2.15)
3. ψ_{MLLR} – MLLR adaptation of UBM was performed given each speaker's feature vectors, only one (global) MLLR matrix was computed yielding a $(D+1) \times D = 41 \times 40 = 1640$ dimensional SV; the mapping is specified in Section (2.3.2)

SVM Note that for each speaker's recording only one SV of each type was extracted. SVM is a binary classifier, thus to train a SVM for each speaker a SVM impostor/background set was constructed (one-against-all training) from all SVs extracted from NIST040506. Since the dimensionality of SVs was high enough to be separated by a hyperplane, only a linear kernel was used – the verification consisted only in a scalar multiplication of two vectors – the SV of the hypothesized speaker and the normal vector of the separating hyperplane. SVM was trained using SVMtorch [9].

NAP In order to train a NAP matrix from Section 2.4 corpora SW1, SW2 and SWC were used because of a lot of available sessions (for some speakers more than 20). Details on the corank of the NAP matrix will be given in Section 4.5.

Rank Normalization (RN) Experiments with RN of SVs were also performed. SVs were rank normalized along each dimension, and the rank was determined according to a background population of SVs from SW1, SW2 and SWC. For details of RN see PhD thesis.

i-vectors and PLDA Development data to train the i-vector extractor consisted of corpora NIST040506, SW1, SW2, SWC and FSH (i.e. all the available development data were used). Subsequently, i-vectors from all development corpora were extracted, normalized to unit lengths [17], and used to train a PLDA model (50 iterations were carried out). Details on values of latent dimensions will be given in upcoming sections.

Score Fusion In order to fuse scores of different systems the linear logistic regression from the FoCal tool kit [18] was used. Hence, the combined/fused score was a weighted linear combination of outputs (verification scores) of distinct systems.

Table 4.3: Error rates acquired on female (F) and male (M) parts of NIST08 and NIST10 utilizing the GMM/UBM based system.

	NIST08		NIST10	
	F	M	F	M
EER[%]	12.19	11.79	17.70	15.58

4.4 GMM/UBM: Baseline Experiments

GMMs dominated the task of speaker recognition for more than a decade, the concept was introduced by Reynolds in [1]. Results can be found in Table 4.3.

4.5 SuperVectors (SVs) and SVM

Experiments will be performed on three types of SVs, namely $\boldsymbol{\psi}_{\mathrm{GSV}}$, $\boldsymbol{\psi}_{\mathrm{MLLR}}$, $\boldsymbol{\psi}_{\mathrm{GLDS}}$, which extraction is described in Section 4.3. Several normalizations will be tested and also the dimensionality reduction will be examined.

Normalizations Since the dimensionality of SVs is very high, linear kernels are satisfactory and perform well with SVs [2, 12, 10]. The one often used with GSVs is the Supervector Linear Kernel (SLK) introduced in (2.8). In fact it incorporates the variance normalization, but moreover distinct dimension blocks are weighted. Since each dimension block is associated with a Gaussian in the UBM, the weighting relates to the number of feature vectors aligned to particular Gaussian. In the case of GLDS SVs the diagonal covariance $\boldsymbol{C}_{\mathrm{SV}}$ of SVs was computed on development corpora NIST040506, SW1, SW2, SWC, and each SV was then multiplied by $\boldsymbol{C}_{\mathrm{SV}}^{-1/2}$. And for MLLR based SVs the One-Class Kernel (OCK) given in (2.12) utilizing a block-diagonal normalization matrix was used.

Also Rank Normalization (RN – distribution of values in each dimension of SV was mapped to match uniform distribution) was performed along each dimension of a SV (as proposed in [19]), the rank normalized dimensions where then transformed with an inverse normal cumulative distribution function yielding Gauss Normalized (GN) SVs. Results on NIST08 can be found in Table 4.4.

Interestingly, best performing system is the one with a simple linear kernel ($K = \boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\psi}$ – in Table 4.4 denoted as "no-norm"). Hence, no additional normalization helped. Since all the normalizations are focused on the variance of the SV, obviously the information on the variance is helpful and does not need to be removed. More detailed explanation related to the way SV is constructed can be found in the PhD thesis.

NAP Now the influence of the corank of the NAP matrix will be examined. Several values were tested, the results can be found in Table 4.5 (corank 0 stands for none NAP). The increase in the performance of the speaker recognition system for NIST08 is more evident for male speakers, but error rates decreased for all tested values of

Table 4.4: Equal error rates [%] acquired on female (F) and male (M) parts of NIST08 utilizing SV/SVM system and distinct types of SVs and normalizations.

ψ_{GSV}	no-norm	RN	RN-GN	SLK
F	9.38	9.50	9.53	10.27
M	8.78	8.94	8.73	10.23

ψ_{GLDS}	no-norm	RN	RN-GN	COV
F	11.16	11.81	11.75	11.54
M	9.19	9.45	9.77	9.92

ψ_{MLLR}	no-norm	RN	RN-GN	OCK
F	13.61	14.74	14.74	13.67
M	12.21	12.83	12.78	12.42

Table 4.5: Equal rrror rates [%] acquired on female (F) and male (M) parts of NIST08 and NIST10 utilizing the NAP normalization and SV/SVM based system.

		NAP corank						
		NIST08					NIST10	
ψ_{GSV}		0	64	128	256	512	0	256
F	EER [%]	9.38	8.73	8.94	9.12	9.38	12.17	9.31
M	EER [%]	8.78	7.17	7.17	7.27	7.48	10.63	7.68

ψ_{GLDS}		0	32	64	128	256	0	64
F	EER [%]	11.16	10.09	10.12	10.45	11.25	14.75	11.71
M	EER [%]	9.19	8.05	8.21	8.26	8.88	12.21	9.16

ψ_{MLLR}		0	8	16	32	64	0	16
F	EER [%]	13.61	13.29	13.17	13.55	14.23	16.96	14.38
M	EER [%]	12.21	11.84	11.90	12.05	11.84	14.42	11.37

the NAP corank. Note that one value of the corank was taken and tested on NIST10. Since it is quite hard to predict the behaviour of the system on unseen data higher values of corank were taken (rather than the best performing one) in order to suppress possible undesirable (strong) within-speaker deviations in test data. The decrease of error rates obtained on NIST10 is substantial.

Principal Component Analysis (PCA) We will investigate whether the dimensionality of a SVM model can be further reduced without loosing any information. A SVM model was trained for each recording from corpora SW1, SW2, SWC, FSH and the background population of SVs (negative examples) for the training were taken from NIST040506. In order to find a SVM model subspace PCA was performed – eigenvalue decomposition of the covariance matrix computed from SVM models obtained from the development set. The question whether the dimensionality reduction

Table 4.6: Equal error rates [%] acquired on female (F) and male (M) parts of NIST08 and NIST10 after the dimensionality reduction of SVM models.

		SVM dim					
ψ_{GSV}		full-dim	7000	5000	3000	1000	500
F	NIST08	9.12	9.77	9.53	9.85	10.09	10.48
	NIST10	9.31	9.68	9.49	9.59	9.86	9.95
M	NIST08	7.27	8.05	7.79	7.84	8.10	8.52
	NIST10	7.68	7.79	7.79	7.68	8.21	8.11

ψ_{GLDS}		full-dim	7000	5000	3000	1000	500
F	NIST08	10.12	10.33	10.30	10.42	11.07	11.75
	NIST10	11.71	11.89	11.80	11.98	12.53	13.46
M	NIST08	8.21	8.31	8.31	8.62	9.25	10.03
	NIST10	9.16	9.26	9.26	9.37	10.11	10.63

ψ_{MLLR}		full-dim	1400	1200	1000	600	500
F	NIST08	13.17	13.17	13.20	13.23	13.32	13.61
	NIST10	14.38	14.38	14.38	14.65	14.38	14.47
M	NIST08	11.90	11.84	11.84	11.90	11.90	12.10
	NIST10	11.37	11.37	11.37	11.47	11.79	12.11

preserves the recognition rates is answered in Table 4.6.

Error rates increased a little, but the performance is still very high even if SVM models and SVs were projected to a much lower dimensional subspace. It is obvious that the dimensionality of the SVM model space contains a lot of redundant information, and it would be interesting to utilize a SVM kernel that maps into a lower dimensional space instead to higher in cases when such high dimensional SVs are used.

Note that the contribution of MLLR SVs to the recognition is lower than the contribution of GSVs and GLDS SVs, the causes are: the low dimensionality of MLLR SVs, the fact that only one MLLR transformation matrix was used, and that the UBM was adapted instead of adaptation of an complex HMM based system. Much lower error rates can be acquired when LVCSR system is in use [20, 19].

4.6 PLDA and i-vectors

The crucial problem when proposing a speaker verification system composed of modules (e.g. JFA, PLDA) is that data from a lot of speakers are required, moreover several sessions have to be available for each speaker in order to train a reliable i-vector extractor and a PLDA model. The problem faced in this section will address the question whether distinct speech corpora (SW1, SW2, NIST040506, SWC) should be pooled together and used to train one PLDA model, or if each corpus should be

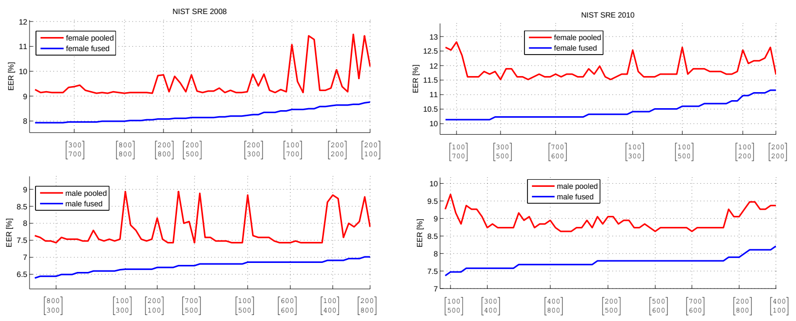

Figure 4.1: Plots depict the comparison of situations when development corpora (NIST040506, SW1, SW2) are pooled and fused for NIST08 and NIST10.

used individually to train a separate PLDA model. In the latter scenario the results are fused at the end. Since often the verification conditions are unknown in advance (e.g. in the Speaker Recognition Evaluations (SREs) organized by NIST and other institutions) we cannot count on the use of one specific speech corpus performing best on the development set. It is more convenient to utilize several corpora, and evaluate the contribution of a PLDA model, trained for each corpora, in the fusion phase.

4.6.1 Development Corpora: NIST040506, SW1, SW2

At first all the development data are pooled together and one PLDA model is trained. Next, one PLDA model is trained for each corpus (hence, three models are trained in total), each system based on one PLDA model is then scored against trials in NIST08. Given true identities of each pair of speakers scored in the trials the logistic linear regression is used to estimate the Fusion Coefficients (FCs) related to each PLDA model. Finally, to prove the validity of learned FCs trials from NIST10 are scored.

The question is which of the systems, the pooled or the fused one, performs better. For this purpose Figure 4.1 was created, where EERs of the fused system (blue line) were sorted, and for each value of EER and latent dimension D_h and D_w for which this value occurred, value of EER of the pooled system (red line) in point $[D_h, D_w]^{\mathrm{T}}$ is plotted. For several values of EER also the dimensions D_h and D_w of latent variables are specified on the x-axis. Note that the fused system performs better in each of the conditions (for each dimension D_h and D_w), moreover variations in error rates related to distinct latent dimensions are lower.

4.6.2 Development Corpora: NIST040506, SW1, SW2, SWC

Now the SWC corpus is added, and the previous experiments will be repeated. Results related to the pooled corpora and to the fused system (fusion coefficients trained again on NIST08) can be found in Figure 4.2. Note that the fused system

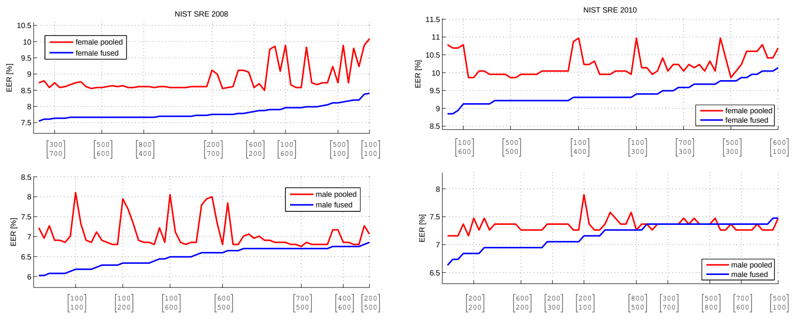

Figure 4.2: Plots depict the comparison of situations when development corpora (NIST040506, SW1, SW2, SWC) are pooled and fused.

does still outperform the pooled one in the case of female speakers, but for male speakers this is no longer true for a substantial amount of values of latent dimensions. However, the fused system does not get worse than the pooled one, and for certain values of D_h and D_w the fused system still performs best. Apparently, having a larger development set leads to more efficient cancelling/averaging of undesirable acoustic deviations in this set.

4.7 Complementarity Analysis

In order to investigate the complementarity of SVM based systems and i-vector based system, outputs (verification scores) of these systems will be fused. For this purpose the logistic linear regression from the FoCal tool kit [18] will be utilized. To train the Fusion Coefficients (FCs) NIST08 will be utilized, and the learned FCs will be then used to fuse outputs of systems trained for NIST10. Let us summarize the main ideas and dissimilarities of methods examined in this thesis:

1. i-vectors combined with a PLDA model are used to find a low dimensional representation of a Supervector (SV) similar to GMM-mean SV (GSV), moreover PLDA decomposes the feature space into speaker- and session-dependent parts
2. i-vectors and PLDA model are generative and do not discriminate between speakers, whereas SVM as a discriminative classifier does; note that even if PLDA is a discriminative model it discriminates between the speaker- and the channel-subspace
3. presented SVs used with SVM incorporate different kinds of information
4. GSV is build from a set of vectors pointing to positions in the feature space with increased concentration of feature vectors; these vectors are concatenated to a high dimensional SV
5. in the case of GLDS the covariance and higher order moments of the whole speaker's data set are extracted

Table 4.7: Equal error rates [%] for different speaker recognition systems and their fusion.

	female		male	
	NIST08	**NIST10**	**NIST08**	**NIST10**
GLDS	10.12	11.71	8.21	9.16
GSV	9.12	9.31	7.273	7.68
MLLR	13.17	14.38	11.90	11.37
PLDA	7.96	9.12	6.49	7.05
GSV-GLDS	8.23	9.31	6.65	7.05
GSV-GLDS-MLLR	8.14	9.22	6.70	7.05
GSV-GLDS-PLDA	6.78	8.48	5.51	5.79
ALL	6.78	8.48	5.30	6.00

6. MLLR is used to transform all the means of a UBM in order to fit given feature vectors, therefore information contained in the MLLR SVs can be thought of as a "model error" (UBM error) given a (spekaer's) feature set

In the fusion systems GSV-NAP-256, GLDS-NAP-64, MLLR-NAP-16 (trained via SVM with a simple linear kernel) performing best in experiments from previous sections will participate along with the fused i-vec/PLDA system trained on NIST040506, SW1, SW2 and SWC, and $D_h = 100$, $D_w = 600$ (they give good results in all four cases male/female and NIST08/NIST10). Results are given in Table 4.7. The fusion of systems is undoubtedly beneficial since error rates decreased in all cases. However, the performance of the MLLR based system is too poor in comparison with the other systems and the fusion does not contribute any further to the recognition performance.

5 Estimation of GMM Statistics on GPU

The Expectation-Maximization (EM) algorithm, in clustering often used also with Gaussian Mixture Models (GMMs), was in [21] identified as one of the top 10 data mining algorithms. GMMs trained via EM are widely used in many state-of-the art recognition and data mining systems. They are of most importance in the speaker recognition. They are utilized in the concept of supervectors and Support Vector Machines (SVMs) and also in Factor Analysis (FA) based systems like JFA and i-Vectors (see Chapter 3). Another usage can be found in speech recognition systems based on Hidden Markov Models with output probabilities described by GMMs [22]. Nevertheless, GMMs are utilized also by biologists and immunologists for counting, sorting, and analyzing cells suspended in a fluid [23]. This chapter is based on the work published in [24].

Focus is laid on GMMs described by diagonal covariance matrices. Only the estimation of GMM statistics is implemented on GPU, rather than the overall estimation of new GMM parameters. The statistics are more general and may be used also in other techniques, e.g. in the adaptation or i-vector extraction discussed in Section 3.1. Note that the estimation process does not involve any approximations, GMM statistics obtained using any of the methods are equal (to some negligible rounding errors).

5.1 Estimation Utilizing CUDA

GPU's CUDA may be seen as a fully parallel system operating with hundreds of threads at once. According to the GPU architecture threads are organized into *thread blocks*. All thread blocks are ordered in an one- or two-dimensional *grid*. Thread blocks are independent of each other (algorithm executed in each of the blocks does not depend on what is going on in other blocks), while threads in each block are allowed to cooperate. All thread blocks execute the same algorithm called a *kernel*. The information on the position in the grid along with the grid dimension is utilized to properly divide input data into smaller independent portions. Each of the data portions is then handled by a separate thread block according to the specified kernel function.

Not all the tasks can be parallelized using only one kernel function since a problem can not always be divided into several fully independent parallel subtasks. More often a result of one subtask depends on a result of a different subtask. However, such tasks may have only a few points where they need to exchange their outcomes. Thus, to parallelize the task one has to employ more kernels. We have proposed 4 kernels

- *$\hat{\gamma}$-kernel* – computes $\hat{\gamma}_{m,t} = \log(\omega_m) + \sum_{i=1}^{D} \log \mathcal{N}(o_{t,i}|\mu_{m,i}, \sigma^2_{m,i})$ for each t, m,
- *$\mathcal{L}$-kernel* – computes overall log-likelihood $\mathcal{L}_t = \log \sum_{m=1}^{M} \omega_m \mathcal{N}(\boldsymbol{o}_t|\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$ for each t,

- *γ-kernel* – normalizes each $\hat{\gamma}_{m,t}$ by $\mathcal{L}_t$ to get a proper Gaussian posterior $\gamma_m(\boldsymbol{o}_t)$,
- *ε-kernel* – estimates first and second moments $\boldsymbol{\varepsilon}_m = \sum_{t=1}^{T} \gamma_m(\boldsymbol{o}_t)\boldsymbol{o}_t$ and $\boldsymbol{\varepsilon}_m^2 = \sum_{t=1}^{T} \gamma_m(\boldsymbol{o}_t)\boldsymbol{o}_t\boldsymbol{o}_t^{\mathrm{T}}$ for each m,

where D is the dimension of a feature vector $\boldsymbol{o}_t$, M is the number of Gaussians in the GMM, ω_m, $\boldsymbol{\mu}_m$, $\boldsymbol{\Sigma}_m$ are weight, mean vector and covariance matrix of Gaussian m, respectively, and $\boldsymbol{\sigma}^2$ is the diagonal of $\boldsymbol{\Sigma}_m$.

A high GPU computing performance can be fully utilized only with proper memory management. Several memory types exist, which significantly differ in their size, access speed and access permission. *Global memory* (GM) has read/write access, has hundreds of mega bytes available and can be accessed from every block and every thread, but the access latency is relatively high. The best performance of GM can be achieved using the *Texture Memory* (TM). TM can be seen as a part of GM, but it is read only and cached, thus the access speed may be significantly faster than in the case of GM. Another type of memory is the *Shared Memory* (SM), which storage size is around kilo bytes, but the access speed is very high (very low latency). SM is visible only for threads in a thread block. In order to make the best of the GPU computing power one has to align the data into Memory-Aligned-Blocks (MABs). The optimal size of a MAB is closely related to the number of threads in a thread block. Number of threads in a block is user dependent, but optimally has to be a multiple of the *warp size*. Warp size is hardware dependent and represents the minimum number of threads in a thread block that run at once (mostly a multiple of 32) – *run in a warp*. For the best performance threads in a warp have to access data in the memory sequentially therefore data in MABs have to be properly organized. Since CUDA supports *X4* data types (e.g. *short4, int4, float4*, etc.), data are stored in the memory in quaternions, and the normalization coefficient of each Gaussian $g_m = \log(\omega_m) - 0.5D\log(2\pi) - 0.5\log|\boldsymbol{\Sigma}_m|$ is precomputed. It should be stated that all the GPU's memory management of feature vectors, model parameters, temporary data (once computed) is assigned to the cached TM (all data are visible to all thread blocks and their threads). However, feature vectors are copied to the faster SM in some kernels.

5.2 Experiments

Experiments were performed on a single EM iteration. Data were taken from NIST SRE 2008, only training data were used for adaptation. In summary, we extracted 3,125,506 (3125.6k) feature vectors of dimension 40. CPU implementation was tested on 2.39 GHz Intel 4 GB RAM PC, the GPU implementation was tested on low-end NVIDIA GeForce GTX 280 video card and the algorithms were developed in CUDA toolkit 3.1.

We have tried to compare the time consumptions also with other implementations. We have tested several freely available implementations, but all of them failed (lack of numerical stability) on our large dataset of high dimensional real data. We have found two recent publications (worth to be mentioned) interested in the GPU implementation of the EM algorithm focusing on GMMs with diagonal covariances, namely a publication by Kumar et al. [25] and a master thesis from Andrew Pangborn [23].

Table 5.1: Comparison given in milliseconds of different implementations for different amounts of training data assuming feature vectors of dimension 32 and GMM with 32 Gaussians.

#samples	Kumar et al.	A.Pangborn	UWB	MATLAB®
153.6k	215.0	51.1	9.25	10936.0
230.4k	264.9	71.1	13.99	16461.0

Experiments performed by Kumar et al. used NVIDIA Quadro FX 5800, which is almost identical to the NVIDIA TESLA C1060 on which the experiments of Andrew Pangborn were performed, and to NVIDIA GeForce GTX 280 on which our experiments were performed. Time consumptions of both implementations were taken from Tab. 5.8 from [23]. In order to compare the implementations to ours we set up same conditions as in [25] and [23]. Hence, we reduced the dimension of our data to 32 and took only 153.6k and 230.4k feature vectors. Tab. 5.1 is the extended table containing also our results denoted as UWB and the CPU reference computed in MATLAB® 7.5.0.342 (R2007b) utilizing the Statistics Toolbox function `gmdistribution.fit()` (only the time spent on estimation of statistics was measured).

As can be seen from Tab. 5.1, the implementation proposed in this chapter outperformed the others. It is more than 5 times faster than A. Pangborn's implementation and more than 20 times faster than Kumar's implementation. The key part of the speed up is the proper memory management of the data adhering to the rules of coalesced access [26]. In addition, data loaded to the kernels are reused as much as possible (higher degree of parallelization), e.g. the log-likelihoods are estimated for several feature vectors and several Gaussians at once in each kernel, the same principle holds for the accumulation kernel. Another important performance related technique lays in the use of the Texture Memory (TM) with *float4* data types for read-only data. Data shared across a thread block or data that are accessed repeatedly should be copied into the Shared Memory (SM) in advance. The mentioned advices are of course well known, but it is quite difficult to integrate them to a specific task.

6 Conclusion

Techniques of automatic speaker recognition developed in the last decade were presented. The main emphasize was laid on the modelling of feature vectors once extracted. Feature vectors are of relatively small dimension (in our case the dimension was 40) and are based on the power spectrum of the speech signal. Feature vectors are mapped to a high dimensional space with the use of a generative model. The task of the generative model is to segment the feature space according to the concentration of feature vectors from the development set localized in specific areas of the feature space. The high dimensional vectors denoted Supervectors (SVs) are composed of various statistics related to feature sets of a speaker, and they can be thought of as a higher level features. For each recording of a speaker only one SV is extracted. Since nowadays large corpora containing hundreds of speakers recorded on several channels (we say that several sessions of a speaker are at hand) are available, information on channel distortions is utilized in order to identify directions in the SV space most responsible for channel and speaker changes. Moreover, since SVs are of substantially high dimension these techniques incorporate also the dimensionality reduction.

The thesis was devoted to a thorough description of methods used in the experiments in a logical sequence. Following problems were solved:

1. *An efficient implementation of the EM estimation algorithm on a Graphics Processing Unit (GPU).* More precisely, the evaluation of EM related statistics was transfered to GPU. Since several thousand hours of speech had to be processed yielding over 10^4 millions of 40 dimensional feature vectors, and since these statistics are required not only when estimating the Universal Background Model (UBM), but are needed also when extracting SVs used with Support Vector Machines (SVMs) and in the i-vector extraction, the speed up was of great importance. Moreover, the EM algorithm and GMM estimation are frequently used also in other fields than the speaker recognition [23, 21].
2. *The influence of normalizations of SVs on the performance of the SVM system.* Surprisingly, the lowest error rates were acquired without any normalizations of SVs. The result was attributed to the pre-normalization of low dimensional feature vectors using the Feature Warping (FW) technique.
3. *An efficient implementation of the PLDA estimation algorithm.* In order to investigate the impact of development data sets on the PLDA modelling analysed in Section 4.6 and performance of the speaker verification system more than 4000 PLDA models were trained for 800 dimensional i-vectors (each training consisted from 50 iterations).
4. *Relation of Nuisance Attribute Projection (NAP) to Factor Analysis (FA) based channel compensation.* Since NAP used with SVM does a channel compensation

and so does the FA model used in JFA/PLDA, the similarities and working principles of both approaches were examined. Both problems were converted to the formulation of Least Squares (LS) and reviewed in the new light of LS yielding interesting conclusions on handling the noise. In simple terms, both do the eigenvector decomposition of a within-speaker covariance matrix, but FA based approach does in addition scale the directions according to the estimated noise level.

5. *The influence of the size of development sets and fusion of PLDA decompositions of several total variability spaces (composed from i-vectors from individual development corpora) on the performance of the speaker verification.* It was shown that if enough data to train a reliable PLDA model are available then it is more convenient to train one PLDA model for each development corpus, and let a fusion algorithm assign a weight to each verification score related to each model. If variations in one corpus would be much higher than in other corpora, and in addition data from such a corpus would be inappropriate for given recognition conditions (e.g. telephone speech), the PLDA model trained from pooled corpora could notably spoil the recognition.
6. *Information redundancy in SVM models.* In addition to the dimensionality reduction of SVs via i-vector extraction, also the dimensionality reduction of SVM model was investigated. Obviously, SVM model does contain a lot of redundancies yielding a SVM decision hyperplane of much lower dimension. Such an observation may be of help when proposing a kernel function, which could instead of mapping to higher dimensions utilize some (e.g. non-linear) mapping to a lower dimensional space.
7. *Complementarity of discussed methods.* It was shown that methods used in the experiments do possess a complementary information since the fused system outperformed all systems based on particular methods.

Bibliography (selection)

[1] D. A. Reynolds, “A Gaussian Mixture Modelling Approach to Text-independent Speaker Identification,” Ph.D. dissertation, Georgia Institute of Technology, 1992.

[2] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, “Support Vector Machines using GMM Supervectors for Speaker Verification,” *Signal Processing Letters, IEEE*, vol. 13, no. 5, pp. 308–311, 2006.

[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,” *Acoustics, Speech and Signal Processing. ICASSP Proceedings*, vol. 1, 2006.

[4] P. Kenny, “Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms,” Centre de Recherche Informatique de Montréal (CRIM), Tech. Rep., 2006.

[5] N. Dehak, “Discriminative and Generative Approaches for Long- and Short-term Speaker Characteristics Modeling: Application to Speaker Verification,” Ph.D. dissertation, École de Technologie Supérieure, Université du Québec, 2009.

[6] S. J. D. Prince and J. H. Elder, “Probabilistic Linear Discriminant Analysis for Inferences About Identity,” *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8, 2007.

[7] 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,” *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*, pp. 4828–4831, 2011.

[8] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, 2001.

[9] R. Collobert, S. Bengio, and C. Williamson, “SVMTorch: Support vector machines for large-scale regression problems,” *Journal of Machine Learning Research*, vol. 1, pp. 143–160, 2001, available from: http://bengio.abracadoudou.com/SVMTorch.html [last visited: 16.8.2012].

[10] R. Dehak, N. Dehak, P. Dumouchel, and P. Kenny, “Linear and Non Linear Kernel GMM SuperVector Machines for Speaker Verification,” *Interspeech*, vol. 1, pp. 302–305, 2007.

[11] Z. N. Karam and W. M. Campbell, “A New Kernel for SVM MLLR Based Speaker Recognition,” *Proc. Interspeech 2007, Antwerp, Belgium*, pp. 290–293, 2007.

[12] W. M. Campbell, “Generalized Linear Discriminant Sequence Kernels for Speaker Recognition,” *IEEE International Conference on Acoustics, Speech and Signal Processing.*, vol. 1, 2002.

[13] K. Lee, C. You, T. Kinnunen, and D. Zhu, “Characterizing Speech Utterances for Speaker Verification with Sequence Kernel SVM,” *Proceedings of Interspeech, Brisbane*, pp. 1397–1400, 2008.

[14] L. Machlica and Z. Zajíc, “Factor Analysis and Nuisance Attribute Projection Revisited,” *Interspeech*, 2012.

[15] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. Springer, 2007.

[16] M. E. Tipping and C. M. Bishop, “Mixtures of Probabilistic Principal Component Analysers,” *Neural Computation 11*, vol. 2, pp. 443–482, 1999.

[17] D. Garcia-Romero and C. Y. Espy-Wilson, “Analysis of I-vector Length Normalization in Speaker Recognition Systems,” *Interspeech*, pp. 249–252, 2011.

[18] N. Brümmer, “FoCal: Tools for Fusion and Calibration of Automatic Speaker Detection Systems,” 2006, available from: http://sites.google.com/site/nikobrummer/focal [last visited: 16.8.2012].

[19] A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, "MLLR Transforms as Features in Speaker Recognition," *Proc. Eurospeech, Lisbon*, pp. 2425–2428, 2005.

[20] N. Scheffer, Y. Lei, L. Ferrer, S. R. I. International, and M. Park, "Factor Analysis Back Ends for MLLR Transforms in Speaker Recognition," *Interspeech*, no. August, pp. 257–260, 2011.

[21] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 Algorithms in Data Mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2007.

[22] J. Vaněk, J. Trmal, and J. Psutka, "Optimization of the Gaussian Mixture Model Evaluation on GPU," *Interspeech*, pp. 1737–1740, 2011.

[23] A. D. Pangborn, *Scalable data clustering using GPUs.* Rochester Institute of Technology, Master Thesis, 2010.

[24] L. Machlica, J. Vaněk, and Z. Zajíc, "Fast Estimation of Gaussian Mixture Model Parameters on GPU using CUDA," *The 12th International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp. 167–172, 2011.

[25] N. Kumar, S. Satoor, and I. Buck, "Fast Parallel Expectation Maximization for Gaussian Mixture Models on GPUs Using CUDA," pp. 103–109, 2009.

[26] "NVIDIA CUDA TM, NVIDIA CUDA C programming guide version 3.2, 11.9.2010."

Authored and Co-authored Works

1. L. Machlica, Z. Zajíc, “Analysis of the Influence of Speech Corpora in the PLDA Verification,” *Lecture Notes in Computer Science*, 2012.
2. L. Machlica, Z. Zajíc, “Factor Analysis and Nuisance Attribute Projection Revisited,” *Interspeech*, 2012.
3. Z. Zajíc, L. Machlica, L. Müller, “Initialization of Adaptation by Sufficient Statistics Using Phonetic Tree,” *IEEE 11th International Conference on Signal Processing (ICSP)*, 2012.
4. Z. Zajíc, L. Machlica, L. Müller, “Bottleneck ANN: dealing with small amount of data in shift-MLLR adaptation,” *IEEE 11th International Conference on Signal Processing (ICSP)*, 2012.
5. L. Machlica, J. Vaněk, Z. Zajíc, “Fast Estimation of Gaussian Mixture Model Parameters on GPU using CUDA,” *The 12th International Conference on Parallel and Distributed Computing, Applications and Technologies*, 2011.
6. Z. Zajíc, L. Machlica, L. Müller, “Initialization of fMLLR with Sufficient Statistics from Similar Speakers,” *Lecture Notes in Computer Science*, 2011.
7. L. Machlica, Z. Zajíc, L. Müller, “Discriminative adaptation based on fast combination of DMAP and DfMLLR,” *Interspeech*, 2010.
8. L. Machlica, Z. Zajíc, L. Müller, “Robust Statistic Estimates for Adaptation in the Task of Speech Recognition,” *Lecture Notes in Computer Science*, 2010.
9. L. Machlica, J. Vaněk, “UWB System Description for NIST SRE 2010,” *NIST 2010 Speaker Recognition Evaluation Workshop – Odyssey’s satellite*, 2010.
10. S. Marcel, C. McCool, P. Matějka, T. Ahonen, J. Černocký, S. Chakraborty, V. Balasubramanian, S. Panchanathan, C. H. Chan, J. Kittler, N. Poh, B. Fauve, O. Glembek, O. Plchot, Z. Jančík, A. Larcher, C. Lévy, D. Matrouf, J.-F. Bonastre, P.-H. Lee, J.-Y. Hung, S.-W. Wu, Y.-P. Hung, L. Machlica, J. Mason, S. Mau, C. Sanderson, D. Monzo, A. Albiol, H. V. Nguyen, L. Bai, Y. Wang, M. Niskanen, M. Turtinen, J. A. Nolazco-Flores, L. P. Garcia-Perera, R. Aceves-Lopez, M. Villegas, R. Paredes, “On the results of the first mobile biometry (MOBIO) face and speaker verification evaluation,” *ICPR’10 Proceedings of the 20th International Conference on Recognizing Patterns in Signals, Speech, Images, and Videos*, 2010.
11. L. Machlica, J. Vaněk, “UWB system description: EVALITA 2009”, *Conference of the Italian Association for Artificial Intelligence*, 2009.
12. Z. Zajíc, L. Machlica, L. Müller, “Refinement Approach for Adaptation Based on Combination of MAP and fMLLR,” *Lecture Notes in Computer Science*, 2009.
13. A. Pražák, Z. Zajíc, L. Machlica, J. V. Psutka, “Fast Speaker Adaptation in Automatic Online Subtitling,” *SIGMAP*, 2009.
14. L. Machlica, Z. Zajíc, A. Pražák, “Methods of Unsupervised Adaptation in Online Speech Recognition,” *SPECOM Proceedings*, 2009.
15. L. Machlica, “Hybrid Modelling in Speaker Recognition,” *PhD Thesis Report*, University of West Bohemia, Pilsen, 2008.

16. Z. Zajíc, L. Machlica, A. Padrta, J. Vaněk, V. Radová, “An Expert System in Speaker Verification Task,” *Interspeech*, 2008.
17. Z. Zajíc, J. Vaněk, L. Machlica, A. Padrta, “A Cohort Methods for Score Normalization in Speaker Verification System, Acceleration of On-line Cohort Methods,” *SPECOM Proceedings*, 2007.
18. L. Machlica, Z. Zajíc, “The Speaker Adaptation of an Acoustic Model,” *The 1st Young Researchers Conference on Applied Sciences*, 2007.
19. L. Machlica, “Ověřování Totožnosti člověka z Nahrávek jeho Hlasu,” *Master Thesis*, University of West Bohemia, Pilsen, 2006.