IDIAP RESEARCH REPORT

IDIAP

Martigny - Valais - Suisse

Comparison of Client Model Adaptation Schemes

Samy Bengio [1]   Johnny Mariéthoz [2]

IDIAP–RR 01-25

August 23, 2001

published in
European Project BANCA Deliverable D22

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 – 27 – 721 77 11
fax +41 – 27 – 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

1 IDIAP, CP 592, 1920 Martigny, Switzerland, bengio@idiap.ch
2 IDIAP, CP 592, 1920 Martigny, Switzerland, marietho@idiap.ch

IDIAP Research Report 01-25

Comparison of Client Model Adaptation Schemes

Samy Bengio Johnny Mariéthoz

August 23, 2001

published in
European Project BANCA Deliverable D22

Contents

Chapter 1

Introduction

In the context of the European project BANCA on Biometric Access Control for Network and e-Commerce Applications, this report gives an overview of the current state-of-the-art client model adaptation methods used in speaker verification.

This chapter first introduces the main problem of identity verification and more specifically the statistical framework used in speaker verification.

In chapter 2, we present the Gaussian Mixture Model as it is the most used model in text independent speaker verification, the framework in which BANCA is working, then give the general method to train such model when enough data is available, that is, by using the Maximum Likelihood principle and training using the Expectation-Maximization algorithm.

Then, in chapter 3, we present two of the most used adaptation methods, namely the Maximum Likelihood Linear Regression method and the Bayesian Maximum A Posteriori method.

Finally, in chapter 4, we compare these adaptation method on two different and well-known benchmark databases: the XM2VTS audio-visual database (where we only used the speech part) and the NIST database used in the 1999 NIST Speaker Recognition Evaluation.

1.1 Identity Verification

The goal of an *automatic identity verification system* is to either accept or reject the identity claim made by a given person. Such systems have many important applications, such as access control, transaction authentication (in telephone banking or remote credit card purchases for instance), voice mail, or secure teleworking.

A good introduction to identity verification, and more specifically to biometric verification can be found in [21].

An identity verification system have to deal with two kinds of events: either the person claiming a given identity is the one who he claims to be (in which case, he is called a *client*), or he is not (in which case, he is called an *impostor*). Moreover, the system may generally take two decisions: either *accept* the *client* or *reject* him and decide he is an *impostor*.

Thus, the system may make two types of errors: *false acceptances* (FA), when the system accepts an *impostor*, and *false rejections* (FR), when the system rejects a *client*.

In order to be independent on the specific dataset distribution, the performance of the system is often measured in terms of these two different errors, as follows:

$$\text{FAR} = \frac{\text{number of FAs}}{\text{number of impostor accesses}}\,, \tag{1.1}$$

$$\text{FRR} = \frac{\text{number of FRs}}{\text{number of client accesses}}\,. \tag{1.2}$$

A unique measure often used combines these two ratios into the so-called *decision cost function* (DCF) as follows:

$$\mathrm{DCF} = \mathrm{Cost(FR)} \cdot P(\mathrm{client}) \cdot \mathrm{FRR} + \mathrm{Cost(FA)} \cdot P(\mathrm{impostor}) \cdot \mathrm{FAR} \tag{1.3}$$

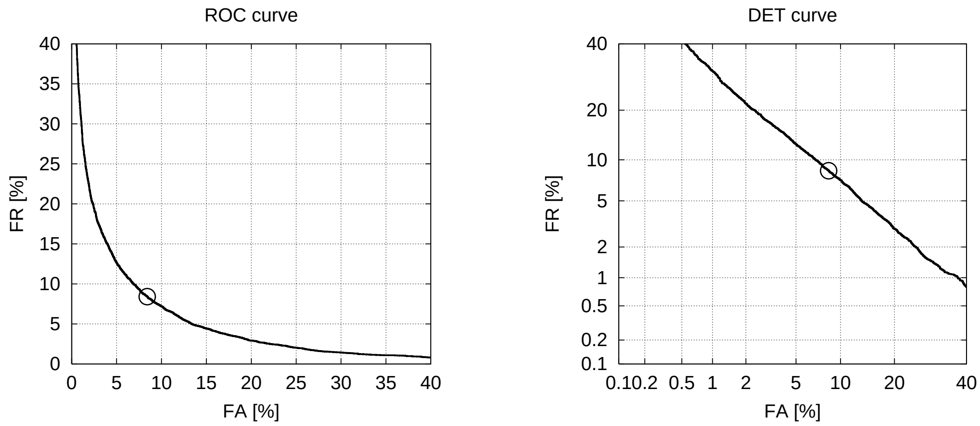
where $P(\text{client})$ is the prior probability that a client will use the system, $P(\text{impostor})$ is the prior probability that an impostor will use the system, Cost(FR) is the cost of a false rejection, and Cost(FA) is the cost of a false acceptance.

A particular case of the DCF is known as the *half total error rate* (HTER) where the costs are equal to 1 and the probabilities are 0.5 each:

$$\mathrm{HTER} = \frac{\mathrm{FAR} + \mathrm{FRR}}{2} \ . \tag{1.4}$$

Most identity verification systems can be tuned using some kind of threshold in order to obtain a compromise between either a small FAR or a small FRR but cannot generally obtain both on a separate validation set.

There is thus a tradeoff which depends on the application: it might sometimes be more important to have a system with a very small FAR, while in other situations it might be more important to have a system with a small FRR. In order to see the performance of a system with respect to this tradeoff, we usually plot the so-called *Receiver Operating Characteristic* (ROC) curve, which represents the FAR as a function of the FRR [19]. More recently, other researchers proposed the DET curve [12], a non-linear transformation of the ROC curve in order to make results easier to compare. The non-linearity is in fact a normal deviate, coming from the hypothesis that the scores of client accesses and impostor accesses follow a Gaussian distribution. If this hypothesis is true, the DET curve should be a line. Figure 1.1 shows examples of typical ROC and DET curves.



(a) ROC curve (b) DET curve

Figure 1.1: Typical example of a ROC curve and its corresponding DET curve. The circle represents the threshold at equal error rate (EER), i.e. when FA=FR.

Whether one uses the ROC or the DET curve, each point of the curves corresponds to a particular decision threshold that should be determined specifically for a given application. For instance, some applications cannot afford a lot of false alarms and are thus ready to have a higher false rejection rate. A typical threshold chosen is the one that minimizes the HTER (1.4) or its generalized version, the DCF (1.3). Another typical threshold chosen is the one that reaches the *Equal Error Rate* (EER) where FAR=FRR on a given validation set.

1.2 Statistical Speaker Verification

State-of-the-art speaker verification systems are based on statistical generative models. We are interested in $P(S_i|\mathbf{X})$ the probability that a speaker S_i has pronounced the sentence $\mathbf{X}$. Using Bayes theorem, we can write it as follows:

$$P(S_i|\mathbf{X}) = \frac{p(\mathbf{X}|S_i)P(S_i)}{p(\mathbf{X})}. \tag{1.5}$$

To decide whether or not a speaker S_i has pronounced a given sentence $\mathbf{X}$, we compare $P(S_i|\mathbf{X})$ to the probability that any other speaker has pronounced $\mathbf{X}$, which we write $P(\bar{S}_i|\mathbf{X})$. The decision rule is then:

$$\text{if } P(S_i|\mathbf{X}) > P(\bar{S}_i|\mathbf{X}) \text{ then } \mathbf{X} \text{ was generated by } S_i. \tag{1.6}$$

Using equation (1.5), inequality (1.6) can then be rewritten as:

$$\frac{p(\mathbf{X}|S_i)}{p(\mathbf{X}|\bar{S}_i)} > \frac{P(\bar{S}_i)}{P(S_i)} = \delta_i \tag{1.7}$$

where the ratio of the prior probabilities is usually replaced by a threshold δ_i since it does not depend on $\mathbf{X}$. Moreover, this threshold is chosen in order to optimize one of the cost function defined in section 1.1, such as the DCF defined in equation (1.3) or the HTER defined in equation (1.4). Taking the logarithm of (1.7) leads to the *log likelihood ratio*:

$$\log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\bar{S}_i) > \log \delta_i = \Delta_i. \tag{1.8}$$

When $p(\mathbf{X}|\bar{S}_i)$ is assumed to be the same for all clients, which is often the case, we replace it by a speaker independent model $p(\mathbf{X}|\Omega)$ where Ω represents the *world* of all the speakers.

To implement this system, we thus need to create a model of $p(\mathbf{X}|S_i)$ for every potential speaker S_i, as well as a *world model* $p(\mathbf{X}|\Omega)$, and then we need to estimate the threshold Δ_i for each client S_i. Note however that this threshold is often the same for every client, and is thus noted Δ.

Chapter 2

GMMs and Maximum Likelihood Method

In order to create the models $p(\mathbf{X}|S_i)$ and $p(\mathbf{X}|\Omega)$, we first need to select a particular form of a parametric probability density distribution and then select its parameters. The probability density distribution have to be rich enough to be able to model correctly the target density distribution but simple enough to be able to select its parameters given a criterion to optimize.

2.1 Gaussian Mixture Models

The most used model, in the context of text-independent speaker verification systems, is the Gaussian Mixture Model (GMM) [18]. In order to use such a model, we make the (false) assumption that the frames of the speech utterance are independent from each other: the probability of a sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ given a GMM with N Gaussians can be computed as follows

$$p(\mathbf{X}) = \prod_{t=1}^{T} p(\mathbf{x}_t) = \prod_{t=1}^{T} \sum_{n=1}^{N} w_n \cdot \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) \tag{2.1}$$

where $\mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$ is a Gaussian with mean $\boldsymbol{\mu}_n \in \mathbb{R}^d$ where d is the number of features and with standard deviation $\boldsymbol{\Sigma}_n \in \mathbb{R}^{d^2}$:

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})\right) \tag{2.2}$$

Note that $\boldsymbol{\Sigma}$ is often forced to be diagonal (then represented by a vector $\boldsymbol{\sigma}$) in order to limit the model size and overcome some estimation problems, but this implies that we make the (often false) hypothesis that all the features are uncorrelated.

2.2 The Maximum Likelihood Principle

Suppose one has a set of observed data $\mathbf{X}$ and wants to select the best probability density distribution to model the observations. Suppose also that the choice of distribution will be made out of a family of distributions $p(\mathbf{X}|\theta)$ which is parameterized by a set of parameters θ.

According to the *Maximum Likelihood* (ML) principle one should select θ such that it maximizes the probability density of the observed data $\mathbf{X}$, that is

$$\hat{\theta} = \arg\max_{\theta} p(\mathbf{X}|\theta). \tag{2.3}$$

In other words, this principle proposes to select the parameters such that the observed data is more likely to occur. Unfortunately, for some families of distributions such as Gaussian Mixture Models (GMMs), selecting the parameters that maximize the likelihood is not straightforward.

2.3 Expectation-Maximization

In 1977, Demptser, Laird and Rubin proposed an algorithm [2], named *Expectation-Maximization* (EM) that could be used to maximize the likelihood of a large family of distributions.

The basic idea was first to observe that for some distributions, if an intermediate variable (called *latent* or *hidden* variable) was introduced in the likelihood function, then the estimation step became easier. Moreover, this hidden variable could also be estimated easily given the data and the current value of the parameters. The derivation of the EM algorithm had thus two steps: first express the log likelihood in terms of the distribution of the hidden variable, then select the parameters that maximize the expected likelihood, and iterate again.

More formally, given the log likelihood $\log p(\mathbf{X}|\theta)$ of an observed dataset $\mathbf{X}$ using the parameters θ, one first introduces a hidden variable $\mathbf{Z}$ and is now interested in the *joint* or *complete* log likelihood $\log p(\mathbf{X}, \mathbf{Z}|\theta)$. Let us introduce the *auxiliary function*

$$Q(\theta|\theta^k) = E_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z}|\theta)|\mathbf{X}, \theta^k] \tag{2.4}$$

which is the expectation, over the hidden variable $\mathbf{Z}$, of the log likelihood of the joint density of the observed data and the hidden variable given the observed data and the parameter set θ^k.

The EM algorithm goes as follows: at each iteration k, the first step, or *expectation step* or *E-step*, consists in computing the expected probability of the hidden variable given the data and the current value of the parameters θ^k, while the second step, or *maximization step* or *M-step*, consists in finding a new set of parameters θ^{k+1} which maximizes the auxiliary function given the expected probability of the hidden variable and the data:

$$\theta^{k+1} = \arg\max_{\theta} Q(\theta|\theta^k) \tag{2.5}$$

In fact, this algorithm converges to a local optimum of the auxiliary function, and it has been shown in [2] that maximizing $Q()$ also maximizes the likelihood of the observed data $p(\mathbf{X}|\theta)$.

2.4 EM for GMMs

In this section, we give the EM update equations when the form of the distribution is a Gaussian Mixture Model (GMM), as defined in equation (2.1).

We first define the latent or hidden variable as the the probability that Gaussian n has generated frame $\mathbf{x}_t$, and we note it as $P(n|\mathbf{x}_t)$. The auxiliary function $Q()$ can then be written as

$$Q(\theta|\theta^k) = \sum_{t=1}^{T}\sum_{n=1}^{N} P(n|\mathbf{x}_t) \cdot \log w_n + \sum_{t=1}^{T}\sum_{n=1}^{N} P(n|\mathbf{x}_t) \cdot \log \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n). \tag{2.6}$$

During the E-step, the posterior probability of the hidden variable is estimated using Bayes theorem as follows:

$$P(n|\mathbf{x}_t) = \frac{p(\mathbf{x}_t|n) \cdot P(n)}{p(\mathbf{x}_t)} \tag{2.7}$$

where $p(\mathbf{x}_t|n)$ is the evaluation of the Gaussian n for $\mathbf{x}_t$, $p(\mathbf{x}_t)$ is the evaluation of the mixture of Gaussians for $\mathbf{x}_t$ and $P(n)$ is the weight w_n of Gaussian n in the mixture.

Then, during the M-step, the parameters of the model are updated as follows:

$$\hat{\boldsymbol{\mu}}_n = \frac{\sum_{t=1}^{T} P(n|\mathbf{x}_t) \cdot \mathbf{x}_t}{\sum_{t=1}^{T} P(n|\mathbf{x}_t)} \tag{2.8}$$

$$\hat{\boldsymbol{\sigma}}_n^2 = \frac{1}{d} \frac{\sum_{t=1}^{T} P(n|\mathbf{x}_t) \cdot \|\mathbf{x}_t - \hat{\boldsymbol{\mu}}_n\|^2}{\sum_{t=1}^{T} P(n|\mathbf{x}_t)} \tag{2.9}$$

$$\hat{w}_n = \frac{1}{T} \sum_{t=1}^{T} P(n|\mathbf{x}_t) \tag{2.10}$$

where $\boldsymbol{\sigma}_n$ is the diagonal standard deviation ($\boldsymbol{\Sigma}_n(i,i) = \boldsymbol{\sigma}_n(i)$). This process is then repeated iteratively and is guaranteed to converge to a local maximum of the likelihood $p(\mathbf{X}|\theta)$.

Chapter 3

Client Model Adaptation Methods

In some applications, it is often difficult to collect enough data in order to train successfully a model using the maximum likelihood principle. This is the case in most identity verification frameworks, where the clients are usually not ready to stay many hours in front of an acquisition system, hours often partitioned into multiple sessions in order to account for the variability of the voice through time. On the other hand, it might be easier to collect a small amount of training data for a lot of persons. This is usually what is done in order to create the *world model*, hence, using the maximum likelihood (ML) principle to estimate its parameters.

In order to overcome to lack of training data for each particular client, many researchers have proposed the use of *adaptation methods*, where one first trains a *world model* using ML and then *adapts* it for each client separately using his own training material.

This general methodology have already been applied in other domains than speaker verification, such as *Automatic Speech Recognition* (ASR) where it generally yielded very good performance (see for instance [7] or [14]). Some authors have even studied the case where the adaptation is done incrementally in parallel with the real use of the system [3].

Many different strategies can be used to adapt a generic model using a small training set. In this chapter, we present two of the most used adaptation methods in the speaker verification domain, namely the *Maximum Likelihood Linear Regression* method and the Bayesian *Maximum A Posteriori* method.

3.1 Maximum Likelihood Linear Regression

The Maximum Likelihood Linear Regression (MLLR) method [8] is an adaptation method that has been proposed for Hidden Markov Models (HMMs) with emission distribution modeled using Gaussian Mixture Models (GMMs). Hence, it is also applicable to plain GMMs, which is similar to an HMM with only one state, connected to itself.

The main idea is to constrain the means of the Gaussians of a given client GMM to be linear combinations of the means of the corresponding Gaussians of the world model. Moreover, all the other parameters of the model, such as the standard deviations and the weights, are kept fixed and equal to their corresponding value in the world model.

More formally, given an already trained GMM of the world model with N Gaussians and weights w_{n_w}, means $\boldsymbol{\mu}_{n_w}$ and standard deviations $\boldsymbol{\Sigma}_{n_w}$ as defined in equation (2.1), then the corresponding GMM of the client model will have the same weights w_{n_w} and standard deviations $\boldsymbol{\Sigma}_{n_w}$, but with the following new mean estimates $\hat{\boldsymbol{\mu}}_{n_c}$:

$$\hat{\boldsymbol{\mu}}_{n_c} = \mathbf{A}_n \boldsymbol{\mu}_{n_w} + \mathbf{b}_n \tag{3.1}$$

where the matrix $\mathbf{A}_n$ and the vector $\mathbf{b}_n$ are parameters to be found by maximizing the likelihood of the client data. This can be done using a modified version of the EM algorithm already presented for ML. However, the solution requires a matrix inversion for each mean vector and as they may be ill-conditioned due to lack of data, it might be better to use a generalized EM algorithm which does a gradient ascent using the gradient of the auxiliary function with respect to $\mathbf{A}_n$ and $\mathbf{b}_n$. The update equations then become

$$\mathbf{A}_n = \mathbf{A}_n + \lambda \frac{\partial Q}{\partial \mathbf{A}_n} \tag{3.2}$$

$$\mathbf{b}_n = \mathbf{b}_n + \lambda \frac{\partial Q}{\partial \mathbf{b}_n} \tag{3.3}$$

where λ is a learning rate and the partial derivatives are computed as follows:

$$\frac{\partial Q}{\partial \mathbf{A}_n} = \sum_{t=1}^{T} P(n|\mathbf{x}_t) \cdot \frac{\mathbf{x}_t - \mathbf{A}_n \boldsymbol{\mu}_{n_w} - \mathbf{b}_n}{(\boldsymbol{\sigma}_{n_w})^2} \boldsymbol{\mu}_{n_w} \tag{3.4}$$

$$\frac{\partial Q}{\partial \mathbf{b}_n} = \sum_{t=1}^{T} P(n|\mathbf{x}_t) \cdot \frac{\mathbf{x}_t - \mathbf{A}_n \boldsymbol{\mu}_{n_w} - \mathbf{b}_n}{(\boldsymbol{\sigma}_{n_w})^2} \tag{3.5}$$

where $P(n|\mathbf{x}_t)$ is the posterior already defined in equation (2.7) and $\boldsymbol{\sigma}_{n_w}$ is the diagonal of $\mathbf{\Sigma}_{n_w}$.

The main idea behind MLLR is to constrain the client models to be near the world model with only a few parameters to be adjusted, given the small amount of data available for each client. Unfortunately, if MLLR is applied as is, the number of parameters to be updated becomes bigger than with standard ML, since for each mean vector $\boldsymbol{\mu}_{n_c}$ of size d, one now have a matrix $\mathbf{A}_n$ of size $d \cdot d$ and a vector $\mathbf{b}_n$ of size d to adjust, which is apparently not a good idea.

Hence, the second important idea of MLLR is to *tie* or *cluster* some Gaussians together in order to force them to share the same matrix $\mathbf{A}$ and vector $\mathbf{b}$.

3.1.1 Gaussian Clustering

Many algorithms can be used to cluster the Gaussians in order to reduce the number of parameters. One such algorithm is the *K-means clustering algorithm* [9, 15] where one finds the K centers (clusters) which minimize the Euclidean distance between every Gaussian mean and its nearest center (cluster). Note however that this algorithm needs to select beforehand the number K of clusters.

Another solution, which has been used in the experiments reported in the present document, is the *regression class tree* [4]. This method grows dynamically a binary tree as follows. First the tree contains only one node which mean is equal to the mean of all the Gaussian means and which contains all the Gaussians. Then an iterative splitting method is applied, where a given node is splitted into two children which means are initialized to the mean of the parent node perturbed in opposite directions by a fraction of the variance of the data. Each Gaussian of the parent node is then assigned to the nearest children in the Euclidean space. Afterward, the mean of each children node is recomputed as the mean of the Gaussian means assigned to it.

The process is repeated until each terminal node contains only one Gaussian. Once the tree is fully grown, the clustering is then performed as follows. One first decides a minimum number of observations m a cluster should have and computes for each Gaussian the number of observations which are the nearest to it, in the likelihood sense. Then, each Gaussian is assigned to the deepest node in the tree that contains it and have at least m observations associated to its Gaussians. The nodes containing less than m observations are thus removed from the tree and the other ones correspond to the final clusters.

3.2 Maximum A Posteriori

The Bayesian *Maximum A Posteriori* (MAP) principle [5] differ from ML in that MAP assumes that the parameters θ of the distribution $p(\mathbf{X}|\theta)$ to estimate is also a random variable which has a prior distribution $p(\theta)$.

The posterior probability density of θ given a set of observations $\mathbf{X}$ can thus be written using the Bayes rule as follows:

$$p(\theta|\mathbf{X}) = \frac{p(\mathbf{X}|\theta) \cdot p(\theta)}{p(\mathbf{X})}. \tag{3.6}$$

The MAP principle states that one should select as an estimate of θ the value that maximizes the posterior probability density of θ, that is:

$$\begin{aligned} \hat{\theta} &= \arg\max_{\theta} p(\theta|\mathbf{X}) \\ &= \arg\max_{\theta} p(\mathbf{X}|\theta) \cdot p(\theta). \end{aligned} \tag{3.7}$$

Note that when there is no knowledge about θ, this is equivalent to assuming a non-informative prior, which leads to $p(\theta) =$ constant, hence equation (3.7) reduces to equation (2.3) which is the ML principle.

Given this formulation, one still needs to specify a correct prior $p(\theta)$ and develop new update equations for a given model under this prior. In [5], the authors propose some solutions to these problems for Gaussian Mixture Models. They suggest that the correct prior for the weights of the mixture should be a *Dirichlet* density [6], whereas the correct prior for the means and standard deviations of each individual Gaussian should be a *normal-Wishart* density [1]. The authors then give the full EM derivation for this case and give the new update equations for the weights, means and standard deviations of the GMMs.

Using MAP for client model adaptation usually means that the prior for the parameters of a client model will be represented by the world model parameters. In this context, experiments reported for instance in [17] show that it is not necessary to adjust the weights and the standard deviations, but only the means of the Gaussians. The general update equation given in [5] is then

$$\hat{\boldsymbol{\mu}}_{n_c} = \frac{\alpha_n \boldsymbol{\mu}_{n_w} + \sum_{t=1}^{T} P(n|\mathbf{x}_t)\mathbf{x}_t}{\alpha_n + \sum_{t=1}^{T} P(n|\mathbf{x}_t)} \tag{3.8}$$

where α_n is Gaussian dependent and is chosen by cross-validation.

In [17], the authors propose a modified version of this update equation to give a better explanation of α_n, and which they used during the 1999 NIST Speaker Verification Evaluation:

$$\hat{\boldsymbol{\mu}}_{n_c} = \alpha_n \boldsymbol{\mu}_{n_w} + (1 - \alpha_n)\frac{\sum_{t=1}^{T} P(n|\mathbf{x}_t)\mathbf{x}_t}{\sum_{t=1}^{T} P(n|\mathbf{x}_t)} \tag{3.9}$$

where α_n is now computed as follows:

$$\alpha_n = 1 - \frac{\sum_{t=1}^{T} P(n|\mathbf{x}_t)}{r + \sum_{t=1}^{T} P(n|\mathbf{x}_t)} \tag{3.10}$$

where r is a relevance factor selected by cross-validation.

In fact, in recent preliminary experiments done at IDIAP in the framework of BANCA using equations (3.9) and (3.10), we obtained even better results when α_n was a constant α overall Gaussians:

$$\alpha_n = \alpha \tag{3.11}$$

with α chosen by cross-validation.

Chapter 4

Evaluation of Adaptation Methods

The goal of the present document is to evaluate state-of-the-art client model adaptation techniques for text independent speaker verification. In order to assess the adaptation methods proposed in chapter 3, we chose to compare them on two distinct but known databases. We first chose the XM2VTS database [11] and its associated experimental protocol, the *Lausanne Protocol* [10], as this audio-visual database was close to the BANCA framework. But since it was in fact a small and easy database, we also decided to compare the methods on a more difficult and known problem, namely the NIST database used during the 1999 NIST Speaker Recognition Evaluation.

Thus, in this chapter, we first present the general methodology used to train our models, then for each database, we start by a description of the data and the associated protocol. Then we compare two different adaptation methods as well as the more classical Maximum Likelihood method.

4.1 General Methodology

Regardless of the database used, we have followed the exact same methodology to conduct the experiments presented in this chapter. In this section, we present this methodology in order for the reader to be able to understand all the results presented in the following sections.

4.1.1 Preprocessing

First of all, the original waveforms have been sampled every 10ms and then parameterized into *Mel Frequency Cepstral Coefficients* (MFCC) [16], keeping 16 coefficients and their first derivative (also known as delta), as well as the energy together with its first derivative, for a total of 34 features computed each 10ms frame.

Afterward, a *bi-Gaussian method* have been used in order to remove the silence frames from the data. We trained a Gaussian Mixture Model (GMM) with two Gaussians in an unsupervised mode. The hope was that one Gaussian would capture the speech frames while the second would capture the silence frames, since they have quite different characteristics. We then simply removed the frames for which the maximum likelihood was given by the Gaussian corresponding to the silence frames.

4.1.2 Training the Models

Both databases were separated into three subsets: a training set, a development set, and an evaluation set. The training set of each database contained a large amount of data for the world model and a small amount of data for each client.

While the energy and its first derivative were important in order to remove the silence frames, they were not adapted to the task of discrimination between clients and impostors, and they thus were removed from the features after the silence frames have been removed. Hence, the world and client models were trained with 32 features (instead of 34).

In order to train the world model using Expectation-Maximization (EM) and the Maximum Likelihood (ML) principle, we had to decide the number of Gaussians in the GMM. This number represents the *capacity* [20] of the model and should thus be tuned carefully: if there is not enough Gaussians, then the model will not capture all the characteristics of the data; on the other hand, if there are too many Gaussians, then the model will *overtrain* and specialize on the training data instead of the expected future data. We thus used a *K-fold cross-validation* method on the training set in order to select the size of the GMM as well as other potential *hyper-parameters* that gave the best expected likelihood of the data. One such hyper-parameter is the so-called *v-floor* which represents the minimal value that the variances of the Gaussians of the model can take. This appears to be a very important hyper-parameter and should be tuned carefully. In fact, in order to select only one *v-floor* instead of one per dimension (32), the minimal variance of each dimension was set to be equal to the global variance in this dimension divided by a global factor. This global factor is the *v-floor*.

After the correct size and *v-floor* of the model has been chosen, we retrained the world model using the whole training data.

In order to train the client models, either using the classical ML method or one of the adaptation methods, some *hyper-parameters* had to be selected: for ML, one had to decide the number of Gaussians in the client models; for MAP, one had to decide the α factor between the world and the client model (see equations (3.9) and (3.11)); for MLLR, one had to decide the clustering factor that forced the Gaussians to share their linear regression parameters $\mathbf{A}$ and $\mathbf{b}$. In any case, we used the same methodology: for each value of the hyper-parameter to tune, we trained the client models using the training data available for each client. We then selected the best value of the hyper-parameter as the one that optimized the Equal Error Rate (EER) on the separate development set.

Finally, we report in the following sections the results obtained on the evaluation set. Hence, these results are unbiased as they have not been used for any purpose during the development of the models.

4.2 The XM2VTS Database

4.2.1 Database Description

The XM2VTS database contains synchronized image and speech data as well as sequences with views of rotating heads. The database contains four recording sessions of 295 subjects taken at one month intervals. On each session, two recordings were made, each consisting of a speech shot and head rotation shot (Figure 4.1). The speech shot consisted of frontal face and speech recordings of each subject during the pronunciation of a sentence. During the rotating head shot, the subject was asked to rotate his/her head from center to left to right to center; then to rotate his/her head up and then down then back to center. If the subject wore glasses he/she was then asked to remove them for a few seconds.

The database was acquired using a Sony VX1000E digital cam-corder and a DHR1000UX digital VCR. Video was captured at a color sampling resolution of 4:2:0 and 16 bit audio at a frequency of 32 kHz. The video data was compressed at a fixed ratio of 5:1 in the proprietary DV format. In total the database contains approximately 4 TBytes (4000 Gbytes) of data.

When capturing the database the camera settings were kept constant across all four sessions. The head was illuminated from both left and right sides with diffusion gel sheets being used to keep this illumination as uniform as possible. A blue background was used to allow the head to be easily segmented out using a technique such as chromakey. A high-quality clip-on microphone was used to record the speech. One speech shot consisted of three sentences:

1. “0 1 2 3 4 5 6 7 8 9”

2. “5 0 6 9 2 8 1 3 7 4”

3. "Joe took fathers green shoe bench out"

The use of digits was chosen as this corresponds to a typical application scenario of speaker verification. The three sentences were the same for all speakers to allow the simulation of impostor accesses by all subjects. The second digit utterance was chosen to compensate for prosodic and co-articulation effects. The third item was supposed to represent a phonetically balanced sentence.

Figure 4.1: Face images of the same persons during different sessions and shots.

4.2.2 Experimental Protocol

A protocol has been defined [10] to evaluate the performance of vision- and speech-based person authentication systems on the XM2VTS database. The use of a common protocol should allow the comparison of different methods.

The database was divided into three sets: training set, evaluation set, and test set (see Figure 4.2). The training set was used to build client models, while the evaluation set was used to compute the decision (by estimating thresholds for instance, or parameters of a fusion algorithm). Finally, the test set was used only to estimate the performance of different verification algorithms.

The protocol was based on 295 subjects, 4 recording sessions, and two shots (repetitions) per recording sessions. Only the first two digit sequences were used for each shot. The database was randomly divided into 200 clients, 25 evaluation impostors, and 70 test impostors (See [10] for the subjects' IDs of the three groups). Two different evaluation configurations were defined. They differ in the distribution of client training and client evaluation data as can be seen in Figure 4.2. Both the client training and client evaluation data were drawn from the same recording sessions for Configuration I which might lead to optimistic performances on the evaluation set. For Configuration II on the other hand, the client evaluation and client test sets are drawn from different recording sessions which might lead to more realistic results.

The following number of subjects were used:

- Clients: 200
- Impostors - Evaluation: 25
- Impostors - Test: 70

This led to the following statistics (see also Figure 4.2 for the partitions):

- 1. client training examples: Conf. I: 3 per client, Conf. II: 4 per client

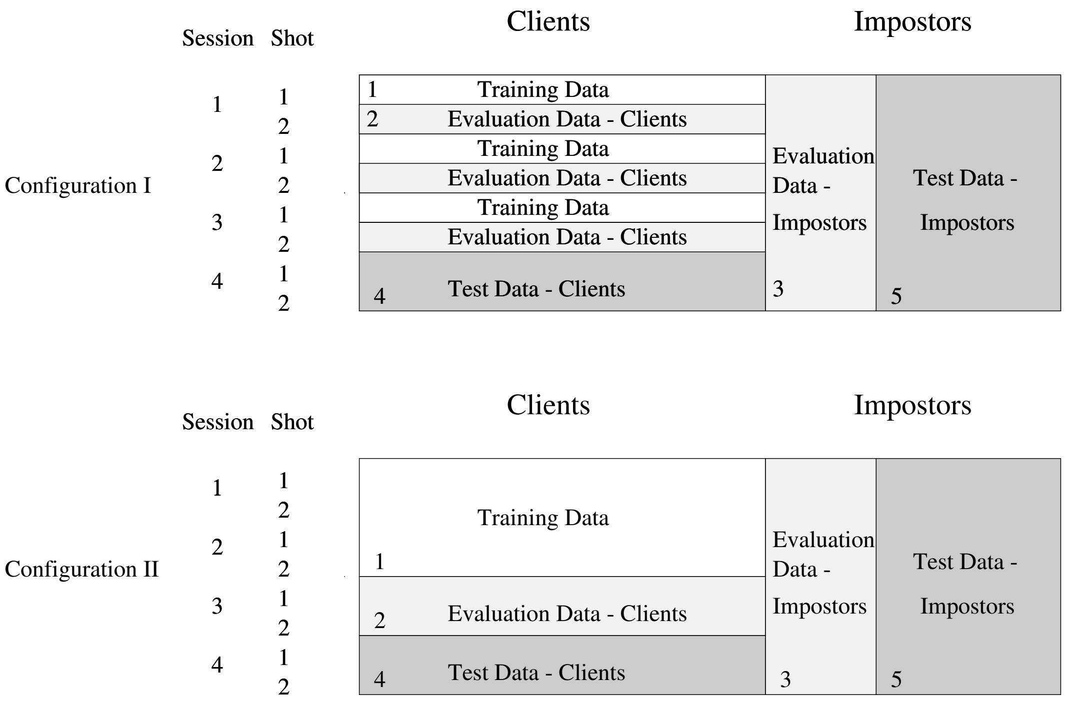


Figure 4.2: Diagramme showing the partitioning of the XM2VTS database according to the protocol Configurations I (top) and II (bottom).

- 2. evaluation - clients: Conf. I - 600, Conf. II - 400
- 3. evaluation - impostors: 40'000 (25 * 8 * 200)
- 4. test client accesses: 400 (200 * 2)
- 5. test impostor accesses: 112'000 (70 * 8 * 200)

4.2.3 Experimental Results

The experiments have been done following the general methodology described in section 4.1. Tables 4.1 and 4.2 summarize the results for configurations I and II respectively. As it can be seen along the HTER columns, the three methods appears quite similar: ML is better on configuration I while MLLR adaptation method is better on configuration II. Given the size of the dataset, the differences between the three methods are probably not really significant.

Note however that in the context on BANCA where the number of bytes needed to represent a client on a smart-card is important and should be as small as possible, the ML method is clearly the best with that respect.

Method	FAR	FRR	HTER	Client Size
Maximum Likelihood	2.66	2.50	2.58	4550
MLLR Adaptation	2.07	3.25	2.66	285120
MAP Adaptation	2.03	3.50	2.76	12800

Table 4.1: Performance of three learning methods on the test set of the speech data from XM2VTS for configuration I. The client size is the average number of floats needed to represent a client.

Method	FAR	FRR	HTER	Client Size
Maximum Likelihood	1.79	2.00	1.89	4550
MLLR Adaptation	1.45	1.50	1.47	285120
MAP Adaptation	1.26	1.75	1.51	12800

Table 4.2: Performance of three learning methods on the test set of the speech data from XM2VTS for configuration II. The client size is the average number of floats needed to represent a client.

In order to compare visually the relative performances of the three methods, Figures 4.3 and 4.4 show the DET curves obtained by the methods on the test set of the speech data from XM2VTS for configurations I and II.

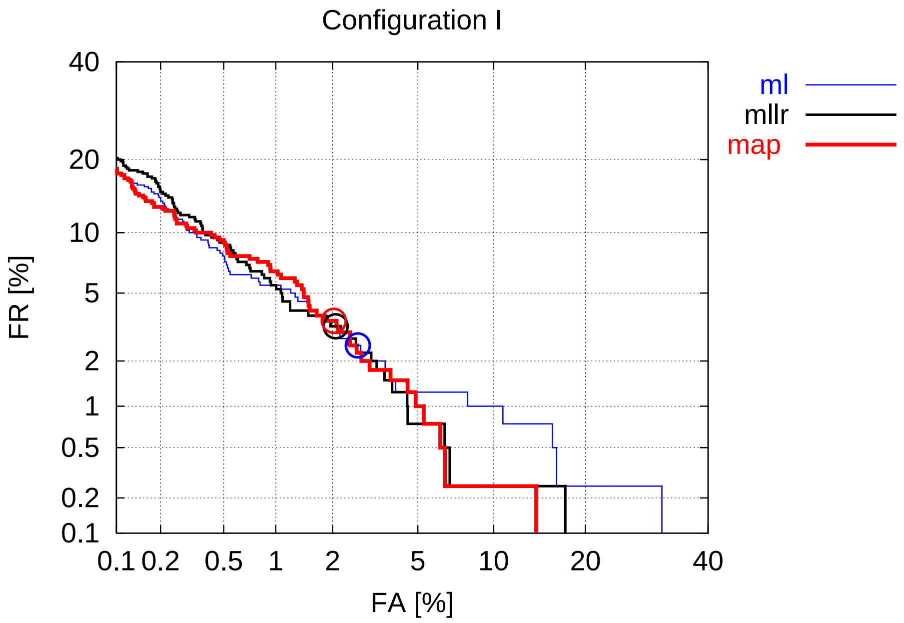


Figure 4.3: Comparison of DET curves of the three methods on the test set of the speech data from XM2VTS for configuration I.

4.3 The NIST Database

4.3.1 Database Description and Experimental Protocol

The NIST database is a subset of a speech only database that was used for the *1999 NIST Speaker Recognition Evaluation*, which was one of the yearly evaluations conducted by the National Institute of Standards and Technology (NIST). An overview of this evaluation as well as the results can be found in [13].

The data for the evaluation came from the Switchboard-2 Phase 3 Corpus collected by the Linguistic Data Consortium (LDC). This corpus consists of 2728 conversions of 5 minute length free speech involving 640 speakers. The participating speakers were mainly college students from the southern part of the United States.

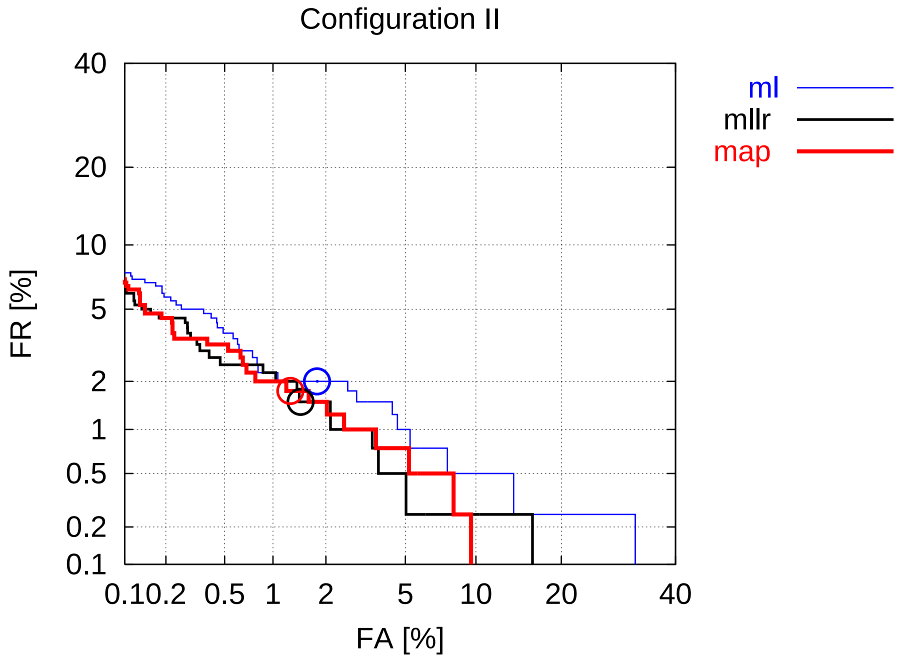


Figure 4.4: Comparison of DET curves of the three methods on the test set of the speech data from XM2VTS for configuration II.

While in the original database two different handsets were used (carbon and electret), in the subset selected for the current report, we only used data from one of the subsets (electret).

As it was done during the contest, we separated the training data into male and female data, in order to create two different world models. The male world model was trained on 137 speakers for a total of 1.5 hours of speech, while the female world model was trained on 218 speakers for a total of 3 hours of speech.

Client data material was separated into two populations: a evaluation and an test set. The evaluation set consisted of accesses of about 2 minutes of telephone speech, while the test set consisted of accesses of between 0 and 1 minute. Each population consisted of 45 males and 45 females. The total number of accesses for each population was around 5000 with a proportion of 10% of true accesses.

4.3.2 Experimental Results

Once again, the experiments have been done following the general methodology described in section 4.1. Table 4.3 summarize the results. As it can be seen along the HTER column, the MAP adaptation method is now much better than the MLLR adaptation method and the ML method, at least on this database.

Note however that in the context on BANCA where the number of bytes needed to represent a client on a smart-card is important and should be as small as possible, the ML method is clearly the best with that respect (but it gives the worst performance, unfortunately). However, if we select a world model with the same number of Gaussians as the one selected for the client models used in the ML method, and use the MAP adaptation method, we obtain a smaller number of parameters than the ML method and about the same performance as the MAP adaptation method.

Again, in order to compare visually the relative performances of the three methods, Figure 4.5 show the DET curves obtained by the methods on the test set of the speech data from XM2VTS for

Method	FAR	FRR	HTER	Client Size
Maximum Likelihood	27.76	23.67	25.71	8320
MLLR Adaptation	22.65	19.25	20.95	327360
MAP Adaptation	16.89	15.71	16.30	16384

Table 4.3: Performance of three learning methods on the test set of the NIST database. The client size is the average number of floats needed to represent a client.

configurations I and II.

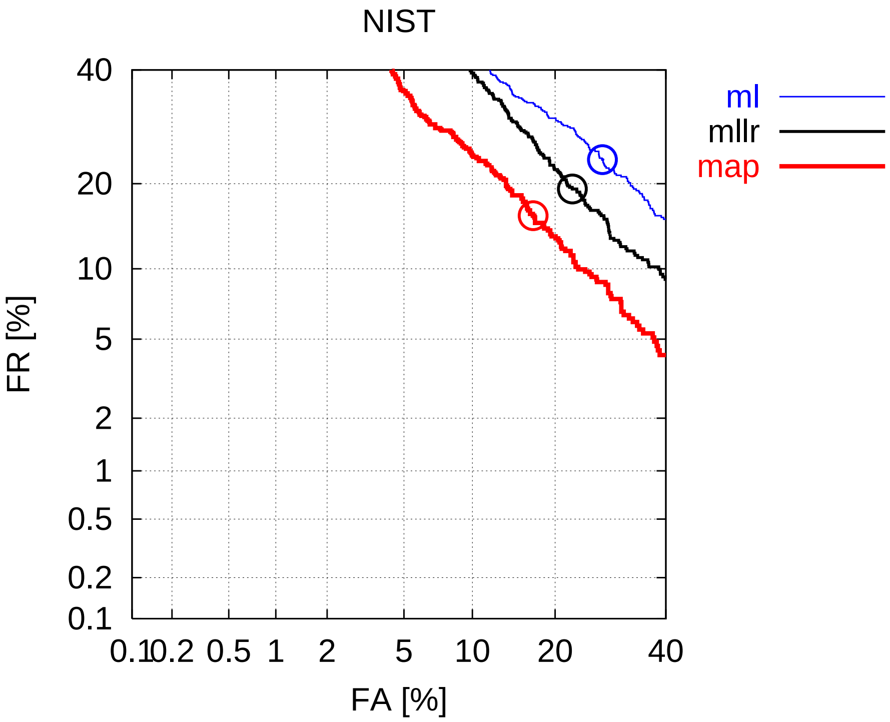


Figure 4.5: Comparison of DET curves of the three methods on the test set of the NIST database.

Chapter 5

Conclusion

In this report, we have presented an overview of state-of-the-art client model adaptation for statistical speaker verification systems, based on Gaussian Mixture Models. We have first described the classical Maximum Likelihood framework, then explained why an adaptation method was required in speaker verification as the size of the training data for client models was usually small compared to the size of the training data available for the world model.

Two different adaptation methods were then presented, namely the Maximum Likelihood Linear Regression (MLLR) method and the Bayesian Maximum A Posteriori (MAP) method. In the MLLR method, the client models are estimated using all the parameters of the world model except that the means are a linear combination of the corresponding means in the world model. The MAP method is derived from a Bayesian perspective, and the means of the Gaussians of the client models are estimated as a weighted sum of the means of the corresponding Gaussians in the world model and the mean of the observed data.

Some experimental results have been presented using two different databases, namely the XM2VTS audio-visual database and the NIST database used in the 1999 NIST Speaker Recognition Evaluation. For each database, the two adaptation methods have been compared, as well as the more classical Maximum Likelihood method.

The analysis of the comparative results of the three methods showed that the MAP adaptation method appears to yield better generalization performance than MLLR and ML on the more realistic NIST database, while the results on XM2VTS are similar for all methods. Moreover, MAP is also much easier to implement than MLLR and needs less parameters. However, if the size of the model is important (as it might be the case in the context of BANCA), then ML needs usually much less space to represent each client and could then be considered.

Bibliography

[1] M. DeGroot. *Optimal Statistical Decisions.* McGraw-Hill, New York, 1970.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B*, 39:1–38, 1977.

[3] C. Fredouille, J. Mariéthoz, C. Jaboulet, J. Hennebert, C. Mokbel, and F. Bimbot. Behavior of a bayesian adaptation method for incremental enrollment in speaker verification. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2000.

[4] M. Gales. The generation and use of regression class trees for MLLR adaptation. Technical Report TR 263, Cambridge University Engineering Department, 1996.

[5] J. L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate gaussian mixture observation of markov chains. In *IEEE Transactions on Speech Audio Processing*, volume 2, pages 291–298, April 1994.

[6] N. L. Johnson and S. Kotz. *Distribution in Statistics.* John Wiley and Sons, New York, 1972.

[7] C.-H. Lee, C.-H. Lin, and B.-H. Juang. A study on speaker adaptation of the parameters of continuous density hidden markov models. *IEEE Transactions on ASSP*, 39(4):806–814, 1991.

[8] C. Leggetter and P. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density HMMs. *Computer Speech and Language*, 9:171–185, 1995.

[9] S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[10] J Lüttin. Evaluation protocol for the the XM2FDB database (lausanne protocol). Technical Report COM-05, IDIAP, 1998.

[11] J. Lüttin and G. Maître. Evaluation protocol for the extended M2VTS database (XM2VTSDB). Technical Report RR-21, IDIAP, 1998.

[12] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proceedings of Eurospeech'97, Rhodes, Greece*, pages 1895–1898, 1997.

[13] A. Martin and M. Przybocki. The NIST 1999 speaker recognition evaluation - an overview. *Digital Signal Processing*, 10:1–18, 2000.

[14] C. Mokbel, L. Mauuary, L. Karray, D. Jouvet, J. Monné, J. Simonin, and K. Bartkova. Towards improving ASR robustness for PSN and GSM telephone applications. *Speech Communication*, 23(1):141–159, 1997.

[15] J. Moody and C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[16] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition.* Prentice All, first edition, 1993.

[17] Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1–3), 2000.

[18] R. C. Rose and Douglas A. Reynolds. Text-independant speaker identification using automatic acoustic segmentation. In *Procedings or International Conference on Acoustics, Speech, and Signal Processing*, pages 293–296, 1990.

[19] H. L. Van Trees. *Detection, Estimation and Modulation Theory, vol. 1.* Wiley, New York, 1968.

[20] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

[21] P. Verlinde, G. Chollet, and M. Acheroy. Multi-modal identity verification using expert fusion. *Information Fusion*, 1:17–33, 2000.