IDIAP RESEARCH REPORT

IDIAP

Martigny - Valais - Suisse

EVALUATION OF BIOMETRIC TECHNOLOGY ON XM2VTS

Samy Bengio [1] Johnny Mariéthoz [2]

Sebastien Marcel [3]

IDIAP–RR 01-21

JULY 6, 2001

PUBLISHED IN
European Project BANCA Deliverable D71

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
3 IDIAP, CP 592, 1920 Martigny, Switzerland, marcel@idiap.ch

IDIAP Research Report 01-21

Evaluation of Biometric Technology on XM2VTS

Samy Bengio Johnny Mariéthoz Sebastien Marcel

July 6, 2001

published in
European Project BANCA Deliverable D71

Contents

Chapter 1

Introduction to Identity Verification

In the context of the european project BANCA on Biometric Access Control for Network and e-Commerce Applications, this report gives an introduction and an evaluation of the current state-of-the-art biometric technologies, including speaker verification, face verification, as well as fusion algorithms merging both techniques. Some experimental results on the XM2VTS benchmark database are also presented in order to compare these methods.

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 [60].

1.1 Measuring the Quality of an Identity Verification System

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*. In some applications, the system may also decide *not to take a decision* because the confidence on the decision would be too low regarding the risks associated with a bad decision. This case will not be studied in the present document but merits some reflexion.

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:

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

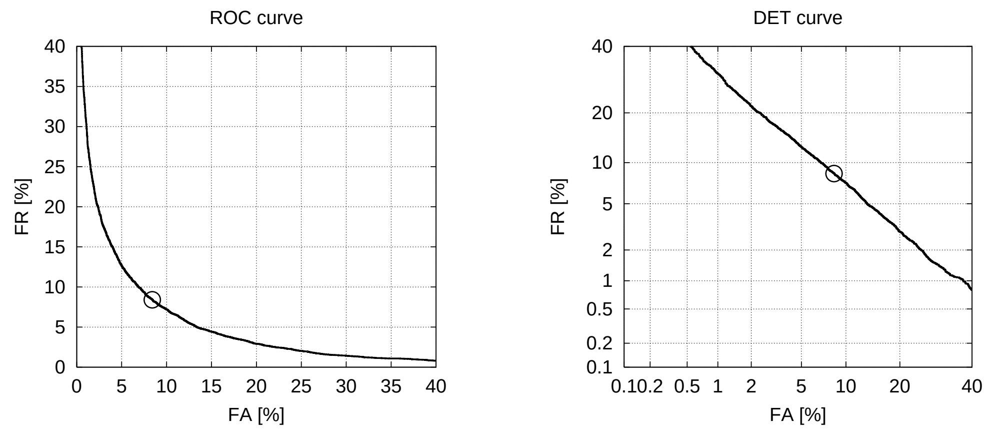
where $P(\text{client})$ is the prior probability that a client will use the system, $P(\text{impostor})$ is the prior probability that an impostor will use the system, Cost(FR) is the cost of a false rejection, and Cost(FA) is the cost of a false acceptance.

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:

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

As it will be clear in the next chapters, 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 [58]. More recently, other researchers proposed the DET curve [35], 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. Note however that many researchers publish results with the threshold is chosen to reach the EER on the test set, which is not realistic as the test set is not supposed to be used to estimate any parameter of a model.

1.2 Biometric Verification

So far, we have explained what is an identity verification system and how to measure its quality but have not yet explained how to construct such a system. Classical systems, already available in many applications, are based on techniques such as passwords, smart cards, and personal identification numbers (PIN). However, these methods have their limits as it is often easy to steal these informations and thus abuse the verification system.

More recently, many researchers have proposed the use of *biometric* characteristics of a person, such as the face, the fingerprint, the signature, the voice, or the eye, as the basic information on which identification systems could be based. Out of these characteristics, some can be said to be *non invasive* as they do not require the person to wear any specific hardware and are thus very user-friendly. In the present study, we will concentrate mainly on two of these characteristics: the voice and the face. In the following two chapters, we thus present the actual state-of-the-art methods used in *speaker verification* as well as *face verification*. Then, in chapter 4, we present fusion algorithms that can merge the results of many verification systems. Finally, in chapter 5, we present some experimental comparisons of unimodal and fusion methods on the same benchmark database, known as the XM2VTS database.

Chapter 2

Speaker Verification

The goal of a speaker verification system is to decide whether a given *speech utterance* has been pronounced by a claimed client or by an impostor. A good introduction to the field can be found in [20]. Different scenarios can take place in this framework: for instance, in *text dependent* applications, the machine knows the lexical content of the utterance used for verification, while in *text independent*, this information is unknown.

This chapter first presents an overview of these scenarios and explains why, given the constraints imposed in the context of the current BANCA European Project, the most appropriate scenario is *text independent*.

Afterward, the chapter presents the preprocessing steps required for most speaker verification systems, and then presents a state-of-the-art text independent speaker verification system, based on generative Gaussian Mixture Models (GMMs).

2.1 Speaker Verification Scenarios

One can think of many ways to verify the identity of a speaker based on its voice. In this section, we briefly introduce two of such scenarios, mainly *text dependent* (which includes *text prompted*) and *text independent* speaker verification. Afterward, a small discussion will be given to explain which scenario is better suited in the context of the BANCA project.

2.1.1 Text Dependent Speaker Verification

In text dependent speaker verification, the system associates a sentence, possibly different, to each client speaker. Hence during an access, a client needs to say his associated sentence, which is known by the system. Therefore, the model created for each speaker can use the lexical information of the sentence to be more client and text specific. One particular case of text dependent speaker verification is known as *text prompted*, where the system *prompts* the potential client with a sentence, which could be different for each access. The main methods used in text dependent speaker verification are:

Dynamic Time Warping: This very simple method has been used in many early speaker verification methods. The idea is to simply record one or more utterances of the associated sentence spoken by a client. Each of these utterances is represented by a sequence of acoustic vectors, as explained later in section 2.2 on preprocessing. Then, during an access, one simply computes the distance between the access utterance and the client utterances, using a dynamic programming method to compare sequences of different sizes. This method is known as *dynamic time warping* [61]. Finally, the access is granted if the computed distance is less than a threshold chosen to optimize FAR and FRR.

Hidden Markov Models: HMMs are statistical models for sequential data that have been used successfully in many applications in artificial intelligence, pattern recognition, speech processing and biological sequence modeling. A good introduction can be found in [45].

The joint probability of a sequence of observations $\mathbf{x}_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ can always be factored as

$$P(\mathbf{x}_1^T) = P(\mathbf{x}_1) \prod_{t=2}^{T} P(\mathbf{x}_t | \mathbf{x}_1^{t-1})$$

which appears intractable in general. However, if we assume that the past sequence can be summarized by a *state variable* q_t, then one can instead write the joint probability of a sequence of observations as:

$$P(\mathbf{x}_1^T) = \sum_{q_1^T} P(\mathbf{x}_1^T, q_1^T)$$

where the sum over q_1^T represents the sum over all possible state sequences $q_1^T = \{q_1, q_2, \dots, q_T\}$ of length T. The joint probability $P(\mathbf{x}_1^T, q_1^T)$ can then be written as

$$P(\mathbf{x}_1^T, q_1^T) = P(q_1) \prod_{t=2}^{T} P(q_t | q_{t-1}) \prod_{t=1}^{T} P(\mathbf{x}_t | q_t) \ .$$

This joint probability is therefore completely specified in terms of *initial state probabilities* $P(q_1)$, *transition probabilities* $P(q_t|q_{t-1})$, and *emission probabilities* $P(\mathbf{x}_t|q_t)$. Since each state variable q_t of the underlying Markov model is not directly observed but is a stochastic function of the observation $\mathbf{x}_t$, such a model is called a *hidden Markov model.* It can be trained to maximize the likelihood of a training set of sequences (Maximum Likelihood criterion), using well known learning algorithms such as Expectation-Maximization (EM) [13], Viterbi [62], or gradient descent [51].

For speaker verification, one first trains an HMM for each client and its associated sentence. The form of the HMM can therefore be adapted to the sentence, as it is usually done in speech recognition. For instance, the emission probabilities are often represented by Gaussian Mixture Models (GMMs) which are explained in more details in section 2.3. The structure of the HMM is usually *left-to-right*, where each state is connected to itself and the next state only. Moreover, the HMM is often sub-divided into smaller HMMs representing sub-units such as phonemes or words, in order to ease the training process and add more lexical constraints to the model. Finally, as it is the case for GMMs and as explained in section 2.3, one can implement adaptation algorithms such as *Maximum A Posteriori* (MAP) in order to create HMM client models starting from a world model for which more data is usually available.

During an access, one computes either the probability of the sentence given the model corresponding to the claimed client (using the forward pass of EM) or the probability of the sentence when going through the best path of the model (using Viterbi). This score is then compared to a threshold in order to decide whether the access is granted or not.

2.1.2 Text Independent Speaker Verification

In text independent speaker verification, the client is not requested to say the same sentence during each access. Hence the only information that can be used by the system are the acoustic caracteristics of the client and not the exact sentence he could say. While in theory, this means that the clients could say anything they have in mind in order to access the system, most clients prefer to have a predefined (or prompted) sentence. However, as the system is not dependent on the lexical content of the sentence, the clients can pronounce them in a very free style way (which is usually not the case with text dependent techniques). On the other hand, it is more difficult to use temporal information to create such systems as no information is known on the specific lexical content pronounced by the clients. In fact, to our knowledge, very few known models for text independent speaker verification use any temporal information (except the usual first and second derivative of the static features). The main methods used in text independent speaker verification are:

Vector Quantization: This method, described for instance in [53], proposes to represent a client model by a set of *representative* acoustic vectors, also called *prototypes*. These vectors are computed using a *vector quantization* algorithm, using a training set of utterances pronounced by a given client. During an access, one simply computes the sum of the distances between each acoustic vector of the access utterance and its closest prototype associated to the claimed client. This score is then compared to a threshold in order to decide whether the access is granted or not.

Sphericity Distance: This simple verification system is described in this section mainly because it has been used in the experiments described in chapter 5. A client is modeled by the covariance matrix $\mathbf{C}$ of the acoustic vectors of the client's training data $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$:

$$\bar{\mathbf{x}} = \frac{1}{T-1}\sum_{i=1}^{T}\mathbf{x}_i, \tag{2.1}$$

$$\mathbf{C} = \frac{1}{T-1}\sum_{i=1}^{T}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^t. \tag{2.2}$$

During an access, the covariance matrix $\mathbf{D}$ is computed over the access utterance. The arithmetic-harmonic sphericity measure $D_{SPH}(\mathbf{C}, \mathbf{D})$ [7] is used as similarity measure between the claimed client and the accessing person:

$$D_{SPH}(\mathbf{C}, \mathbf{D}) = \log\left[\frac{tr(\mathbf{D}\mathbf{C}^{-1})tr(\mathbf{C}\mathbf{D}^{-1})}{m^2}\right], \tag{2.3}$$

where m is the dimension of the acoustic vector and $tr(\mathbf{C})$ the trace of $\mathbf{C}$. This measure is then compared to a threshold in order to decide whether the access is granted or not.

Gaussian Mixture Models: This model represents the current state-of-the-art in text independent speaker verification systems, and will be explain in more details in section 2.3. It is mainly similar to an HMM where the number of states is reduced to only 1 and the emission probability is represented by a mixture of Gaussians (GMMs).

2.1.3 Speaker Verification in the context of the BANCA project

Text dependent speaker verification systems are generally less accepted by users as they require them to memorize a given sentence and to pronounce it very precisely (without adding or removing anything). Text independent systems impose less constraints and let the clients pronounce their (eventually memorized or prompted) sentence in a more free speech approach. It is also important to take into account the fact that text independent systems are technologically easier and faster to implement than text dependent systems. These are the main reasons why we have chosen, in the context of the BANCA project, to concentrate on text independent speaker verification methods.

Moreover, such scenario can enable us to continue to verify the speaker identity during the whole working session (and not just the authentication part) if the session is to be conducted verbally, as it might be the case in BANCA.

Finally, it is worth mentionning that text independent methods are usually faster to implement as they do not require to process the training data manually (such as segmentation, often needed for text dependent models).

However, in theory and taking into account that clients can pronounce correctly their sentence (hence in a very constrained and non free speech way), current state-of-the-art text dependent speaker verification algorithms such as HMMs are usually slightly better in terms of performance than current state-of-the-art text independent algorithms such as GMMs.

In the next sections, we present the main steps required by a state-of-the-art text independent speaker verification system.

2.2 Preprocessing

In the present framework, we call *preprocessing* all the operations made before the creation of the verification models: speech acquisition, parameterization and silence removal. Figure 2.1 shows a schematic view of these operations.

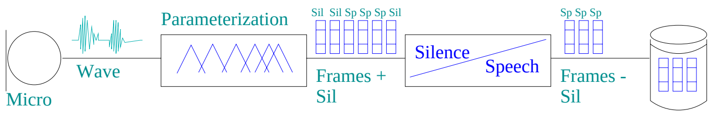


Figure 2.1: Preprocessing steps

2.2.1 Acquisition System

The standard acquisition system is usually a simple microphone which converts air pressure produced by the speaker into an electrical signal. In figure 2.1, the sampling is not represented, but it is supposed to be done by the microphone itself. The sampling rate is typically ranging from 8kHz to 32kHz and the quantization is equal to 16 bits.

2.2.2 Parameterization

Parameterization is the task that transforms the original waveform into a vector a characteristics relevant for a given task. For the same reasons as in automatic speech recognition systems, we want to find some characteristic patterns in the speech signal (formants) [45]. However, in speaker verification, these formants have to be different from speaker to speaker and not from phoneme to phoneme, in order to distinguish two different speaker. As it is not easy to find such patterns in the time domain, some transformations are usually applied to the original signal in order to view it in spectral and cepstral domains.

There are two different categories of features [52] mainly used for speaker verification tasks:

LPCC: This parameterization is based on a linear predictive analysis in order to compute the *Linear Prediction Coefficients* (LPCs). It comes from a speech production model and tries to model the vocal track by a transfer function:

$$H(z) = \frac{A}{1 - \sum_{k=1}^{p} a_k z^{-1}} \tag{2.4}$$

where A is the gain, p is the LPC order, and the a_k are the LPCs. They are selected in order to minimize the error $\delta(t)$ in the equation:

$$x(t) = \sum_{k=1}^{p} a_k x(t-k) + \delta(t). \tag{2.5}$$

To convert the LPCs from the spectral (the $a(k)$) to the cepstral (the $c(i)$) domain, we then use the following equation:

$$c(i) = -a_k(i) - \sum_{k-1}^{i-1} \left(1 - \frac{k}{i}\right) a(k) c(i-k). \tag{2.6}$$

MFCC: These coefficients are based on a speech perception model. This model applies a triangle band filter on a Mel scale [45] in the spectral domain. Based on some fundamental ideas of spectrum analysis, the discrete short-time spectrum of $x(t)$ is defined as:

$$X_l(\omega) = \sum_{t=-\inf}^{l} x(t) h(l-t) \exp(-j\omega t). \tag{2.7}$$

One interpretation of the short-time spectrum is that $X_l(\omega)$ is the Fourier transform of a sequence $x(t)$ that is weighted by a "window" $h(l-t)$.

From this, one can compute the coefficients $\tilde{X}_k$ as the power on each of the K triangle band-pass filters. Then one can compute the so-called mel-frequency cepstrum, $\tilde{c}_n$ as:

$$\tilde{c}_n = \sum_{k=1}^{K} (\log \tilde{X}_k) \cos\left[n\left(k - \frac{1}{2}\right)\frac{\pi}{K}\right], n = 1, 2, ..., L. \tag{2.8}$$

where L is the number of coefficients.

LPCC seems to be more speaker dependent than MFCC, but this should be verified experimentally.
Some additional coefficients such as delta (first derivative), delta-delta (second derivative), and energy or some operations such as cepstral mean substraction may also have an impact on speaker verification and should thus be also tested.

2.2.3 Silence Removal

A very important preprocessing step in speaker verification is to remove the silences from the original signal. This should help in particular to get better trained models and a better estimation of the likelihoods during the decision step. When we train a client model, the goal is to represent the speech distribution of a given client, and the silence parts of the waveforms do not generally contain any interesting speaker information. It is therefore better not to include them during training. Regarding the likelihood estimation, the silence parts have an another annoying effect: they modify the duration normalization. If there is a lot of silence frames, the relative influence of speech frames decreases. Therefore different methods exist in order to detect the silences and remove them. One of them is the *bi-Gaussian method*:

The idea is to train a Gaussian Mixture Model (GMM) [1] with two Gaussians in an unsupervised mode, hoping that one Gaussian will capture the speech frames while the second will capture the

[1] See section 2.3.1 for an introduction on GMMs.

silence frames, since they have quite different characteristics. Afterward, one should simply remove the frames for which the maximum likelihood is given by the Gaussian corresponding to the silence frames.

Note that the parameterization used to discriminate silence and speech can be different from the one used to train the models, because the information relevant for each task is probably different.

2.3 A Statistical Speaker Verification System

State-of-the-art text independent 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:

$$P(S_i|\mathbf{X}) = \frac{p(\mathbf{X}|S_i)P(S_i)}{p(\mathbf{X})}. \tag{2.9}$$

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:

$$\text{if } P(S_i|\mathbf{X}) > P(\bar{S}_i|\mathbf{X}) \text{ then } \mathbf{X} \text{ was generated by } S_i. \tag{2.10}$$

Using equation (2.9), inequality (2.10) can then be rewritten as:

$$\frac{p(\mathbf{X}|S_i)}{p(\mathbf{X}|\bar{S}_i)} > \frac{P(\bar{S}_i)}{P(S_i)} = \delta_i \tag{2.11}$$

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 chapter 1, such as the DCF defined in equation (1.3) or the HTER defined in equation (1.4). Taking the logarithm of (2.11) leads to the *log likelihood ratio*:

$$\log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\bar{S}_i) > \log \delta_i = \Delta_i. \tag{2.12}$$

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.

2.3.1 Training of Client and World Models

In the context of text-independent speaker verification systems, Gaussian Mixture Models (GMMs) are actually the state-of-the-art models [49]. In order to use them, we make the (false) assumption that the frames of the speech utterance are independent from each other: the probability of a sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ given a GMM with N Gaussians can be computed as follows

$$p(\mathbf{X}) = \prod_{t=1}^{T} p(\mathbf{x}_t) = \prod_{t=1}^{T} \sum_{n=1}^{N} w_n \cdot \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) \tag{2.13}$$

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 $\boldsymbol{\Sigma}_n \in \mathbb{R}^{d^2}$:

$$\mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{d}{2}}\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \tag{2.14}$$

Note that $\boldsymbol{\Sigma}$ is often diagonal, making the hypothesis that all the features are uncorrelated.

To train a GMM, the EM algorithm [13] can be used in order to *maximize the likelihood* of the data given the model. The average log likelihood LL of a sequence X is equal to

$$LL = \frac{1}{T}\sum_{t=1}^{T}\log p(\mathbf{x}_t). \tag{2.15}$$

As it is well known that EM is very sensitive to the initialization of the parameters, a simple K-means algorithm [32] is usually performed to select a good candidate parameter set before starting the EM procedure.

An important property of any parametric model such as a GMM is its *capacity* [59], which is related to the number of different functions (or distributions in the case of GMMs) the model can span using its parameter set. The optimal capacity, in order to expect the best performance on new data, depends on the quantity and the nature of the training data. Two factors can influence the capacity of a GMM:

- the number of EM iterations,
- the number of Gaussians in the mixture.

One way to select the optimal capacity is to fix a stopping criterion such as the relative improvement between two consecutive learning iterations and then select the appropriate number of Gaussians using a cross-validation method. Figure 2.2 shows how the expected generalization performance varies as a function of the capacity (such as the number of Gaussians).

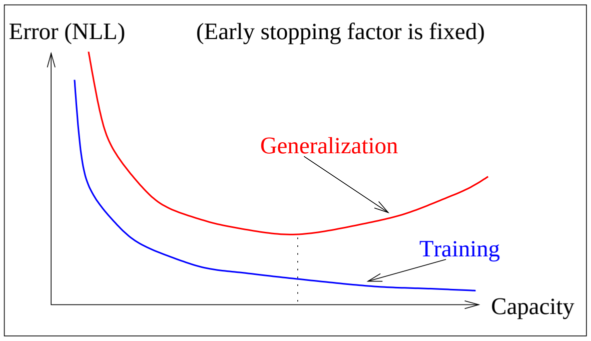


Figure 2.2: Evolution of the negative log likelihood (NLL) with respect to the capacity

A prior on some parameters such as the weights w_n can also be set in order to give an *a priori* distribution on the data, or in most cases, to overcome some zero divisions during training when a Gaussian has not captured any data.

Some singularities can also be present in the data and produce very small variances which can introduce some numerical problems. In this case, it can be necessary to fix a minimum variance value [39]. At least two methods can then be used:

Variance flooring factor: It is just a constant that defines the minimum value a variance can take.

Variance flooring vector: In this case, the minimum value of a variance is a proportion of the global variance estimated on the training data, for each feature independently. The proportion factor is estimated on a development set.

Training a World Model

As explained in section 2.3, in order to normalize the likelihood of a speech utterance given by a client model, we also need an *anti-speaker model*. In the case where the anti-speaker model is the same for all the speakers, it is called a *world model*. To obtain a good world model, it is important to train it using a large number of speakers, of course different from the targeted clients, representing a large amount of speech data, at least 1 hour of speech [47].

After selecting the optimal capacity, the model is trained to maximize the likelihood of the data using the EM algorithm (Figure 2.3).

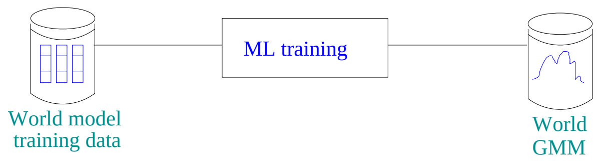


Figure 2.3: World model training step

Training the Client Models

Bayesian *maximum a posteriori* (MAP) adaptation techniques have been successfully applied in speech recognition tasks [21]. Recently it has also been applied with success on speaker verification [47].

The basic principle is to adapt the world model in order to create a client model, instead of starting from random (Figure 2.4). In practical applications, the amount of data available from each client is small (it is usually not possible to ask a client to speak to a machine for many hours in order to create its model). Thus, starting the training process from the world model is the first advantage of the MAP adaptation technique. The second is due to the poor representativity of the speaker data: the adaptation process enables to constrain the client model to be more or less near the world model, using a confidence level one has over the client data (based essentially on the quantity of data for the client). The last advantage is that some Gaussians will not move during the adaptation process and their likelihood will thus have no effect on the computation of the log likelihood ratio, used in the decision step.

The most used adaptation method has been proposed by Reynolds [47] and is inspired by the general MAP framework given by Gauvain and Lee [21]. This method requires to choose an optimal relevance factor (confidence level given to the client data) on a development set. For each Gaussian i and each parameter ρ of this Gaussian, a data-dependent adaptation coefficient α_i^ρ is defined as follows:

$$\alpha_i^\rho = \frac{n_i}{n_i + r^\rho} \tag{2.16}$$

where r^ρ is a relevance factor to select for each parameter ρ and n_i is defined as:

$$n_i = \sum_{t=1}^{T} P(i|\mathbf{x}_t) \tag{2.17}$$

where $P(i|\mathbf{x}_t)$ is the posterior probability of Gaussian i given the frame $\mathbf{x}_t$.

Moreover, further experiments [47] show that it is not necessary to adapt the variances and weights of the GMM: only the means $\boldsymbol{\mu}_i$ have to be adapted using the following adaptation method:

$$\hat{\boldsymbol{\mu}}_i = \alpha_i^\mu \boldsymbol{\mu}_{\omega_i} + (1 - \alpha_i^\mu)\boldsymbol{\mu}_i \tag{2.18}$$

where $\boldsymbol{\mu}_{\omega_i}$ is the mean of the Gaussian i in the world model and $\boldsymbol{\mu}_i$ the mean of Gaussian i of the observed client data.

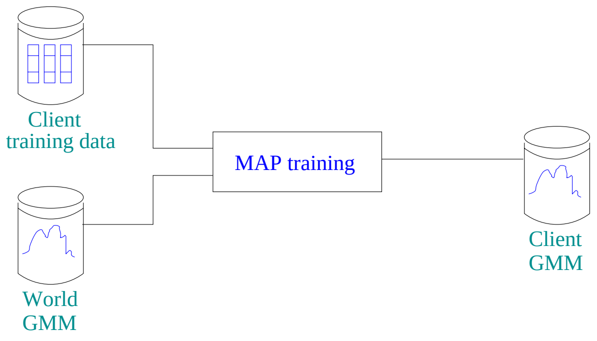


Figure 2.4: Client model adaptation step

Normalization and Estimation of Z-Norm Parameters

In order to be able to compare impostor scores of every client models and thus be able to select a global threshold parameter that will be speaker independent (a unique Δ in equation (2.12)), a normalization step is often required.

The Z-normalization [30] first makes the hypothesis that all impostor scores have been drawn from a Gaussian distribution, then this distribution is estimated and normalized to have a zero mean and a unit standard deviation (such that every impostor scores from every models are now comparable) (Figure 2.5).

Given the log likelihood ratio $LLR(S_i, \mathbf{X})$ of speaker S_i on utterance $\mathbf{X}$ defined as

$$LLR(S_i, \mathbf{X}) = \log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\Omega) \tag{2.19}$$

the normalized log likelihood ratio is then computed as:

$$NLLR(S_i, \mathbf{X}) = \frac{LLR(S_i, \mathbf{X}) - \boldsymbol{\mu}_{S_i}}{\boldsymbol{\sigma}_{S_i}} \tag{2.20}$$

where $\boldsymbol{\mu}_{S_i}$ and $\boldsymbol{\sigma}_{S_i}$ are respectively the mean and standard deviation of the impostor scores, as estimated on an external impostor population (Figure 2.6).

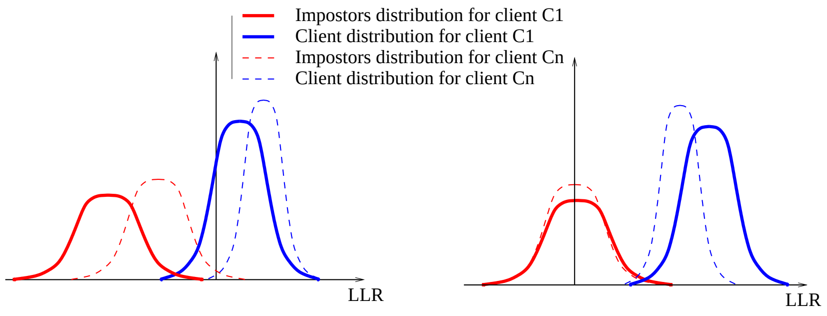


Figure 2.5: The Z-norm principle

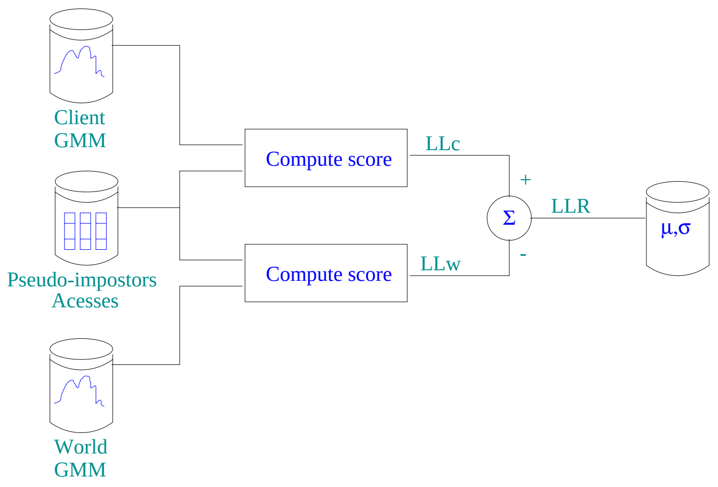


Figure 2.6: Z-norm parameter estimation step

2.3.2 Decision

Estimation of a Global *A Priori* Decision Threshold

Following equations (2.12) and (2.19) as well as conventional Bayesian decision theory [41, 38], the normalized log likelihood ratio $NLLR(S_i, \mathbf{X})$ is compared to a threshold Δ in order to yield a decision on the claimed identity. In general, the threshold Δ is estimated (Figure 2.7) in order to optimize the HTER, as described in chapter 1 and equation (1.4). Note that this cost function changes the relative weight of client and impostor accesses in order to give them equal weight, instead of the one induced by the training data.

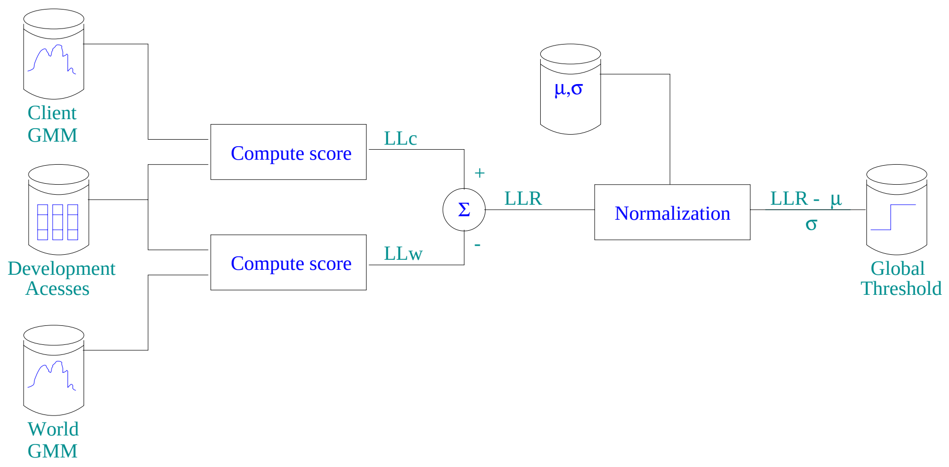


Figure 2.7: Global threshold estimation steps

The HTER can be computed in two different scenarios: on the one hand, the HTER obtained when the threshold Δ_{opt} is set in order to optimize the cost function *a posteriori* on the test set (HTER$_{opt}$) and, on the other hand, the HTER obtained when the threshold Δ_{dev} is optimized *a priori* on a separate development population of clients (HTER$_{dev}$). Note that HTER$_{opt}$ is optimistically biased (HTER$_{opt}$ $\leq$HTER$_{dev}$) and thus less realistic.

Making the Decision

In order to take a decision given a new speech utterance $\mathbf{X}$, one simply applies the following steps (Figure 2.8):

1. Preprocess the speech utterance and produce the sequence $\mathbf{X}$.
2. Compute $P(\mathbf{X}|S_i)$ where S_i is the claimed client, using equation (2.13).
3. Compute $P(\mathbf{X}|\Omega)$ using equation (2.13).
4. Compute the log likelihood ratio $LLR(S_i, \mathbf{X})$ using equation (2.19).
5. Compute the normalized log likelihood ratio $NLLR(S_i, \mathbf{X})$ using equation (2.20).
6. Compare $NLLR(S_i, \mathbf{X})$ to the threshold Δ and take the following decision:

$$\text{if } NLLR(S_i, \mathbf{X}) > \Delta \text{ then } \mathbf{X} \text{ was generated by } S_i. \tag{2.21}$$

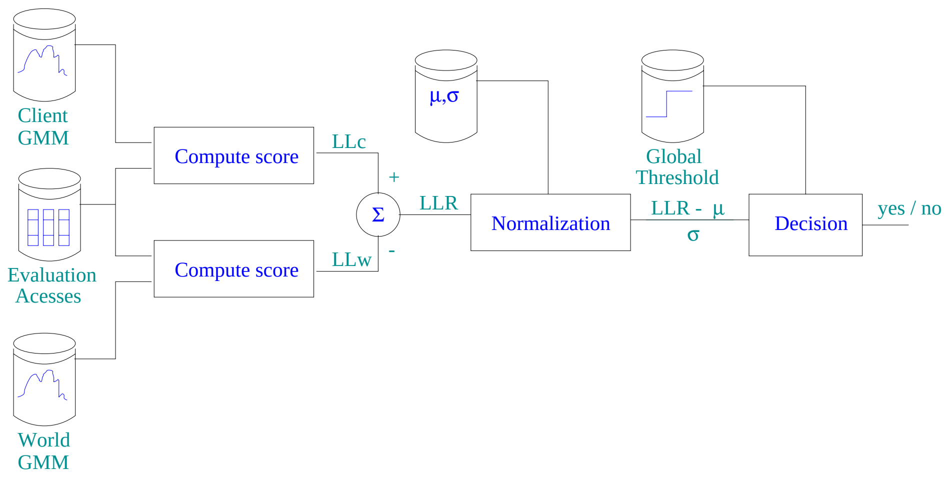


Figure 2.8: Final test step

Chapter 3

Face Verification

Face verification is concerned with validating a claimed identity based on the images of its face, and either accepting or rejecting this identity claim. Face verification should be able to accept subjects, referred to as *clients* using a set of reference images, and should be able to reject subjects, referred to as *impostors*, that were not enrolled in the system (Figure 3.1).

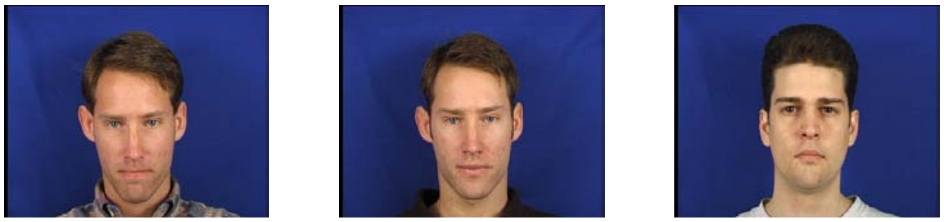


Figure 3.1: Face images of different persons claiming the same identity. From left to right: a reference image of client 001, a candidate image of client 001, a candidate image of impostor 000.

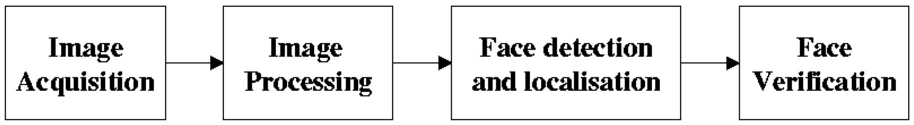


Figure 3.2: Preprocessing steps for face verification.

The whole face verification process is decomposed in several steps (Figure 3.2).

1. **Image Acquisition**: Image acquisition is the very process which grabs the image (color or gray levels) from a camera or a VCR.

2. **Image Processing**: Image processing tries to apply filtering algorithms to the acquired image in order to enhance important features and to reduce the noise.

3. **Face Detection**: Face detection is the fundamental step before the verification procedure. Its goal is to detect and localize an eventual face in a given image. Its reliability and time-response have a major influence on the performance and usability of the whole face verification system.

4. **Face Verification**: Face verification is the final step which uses all reference images of a client to built a client model. Then, a candidate image is compared to the model of the client which he claims to be, in order to decide if the candidate should be accepted or rejected.

The following sections present, first a small introduction to image processing, second a description of the face detection problem and third a non exhaustive list of different state-of-the-art techniques for face verification. Descriptions and explanations are **directly taken from published papers** and detailed references are given.

3.1 Image Processing

Image processing tries to apply filtering algorithms to the acquired image in order to enhance important features and to reduce the noise. In this section, we only consider color images. A image is a set of NxM n-uplets, where N and M are respectively the height and the width of the image and n is the number of bands of the image. If $n = 1$, the image is in gray level, If $n = 3$, the image is in color, Red-Green-Blue (RGB) for instance.

- **Color filtering**: In face verification, we are interested in particular objects, namely, faces. These objects have often a characteritic color which is possible to separate from the rest of the image (Figure 3.3). Numerous methods exist to model the skin color, essentially using Gaussians [64] or Gaussian mixtures [66]. Futhermore, different colorimetric spaces RGB, YUV, HSV or Lab can be used [48].

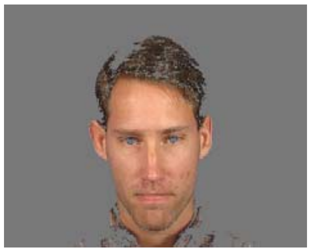


Figure 3.3: Filtering skin color pixels.

Once skin color pixels are filtered, it is possible to extract the bigger object, hopefully the face (Figure 3.4).

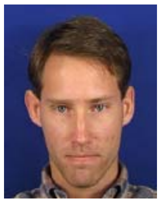


Figure 3.4: Subimage containing the face.

After this extraction, it is possible to work only on gray levels in order to apply enhancing and smoothing algorithms.

- **Enhancement**: The objective of image enhancement is to modify the contrast of the image (Figure 3.5) in order to enhance important features.

- **Smoothing**: The smoothing is a simple algorithm which reduces the noise in the image (after image enhancement for example) by applying a Gaussian to the whole image (Figure 3.6).

Figure 3.5: Gray level subimage of the face enhanced by histogram normalization. From left to right, the original gray level image and the enhanced image.

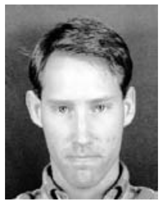


Figure 3.6: Gray level subimage of the face smoothed, after enhancement, by Gaussian convolution.

3.2 Face Detection and Localization

The goal of face detection is to identify and locate human faces in images at different positions, scales, orientations and lighting conditions. This is a challenging problem because of the high degree of variability of faces in images (poses, facial expressions, color) and because of occlusion (glasses, fingers, hands). Face localization is a simplified face detection problem with the assumption that the image contains exactly one face.

In recent years, numerous methods have been proposed to detect faces in gray images [50] [17] [54] [40] and color images [12] [63]. Surveys on face detection methods can be found in [15] [65]. Appearance-based approaches as well as learning-based approaches seem to be better suited for such a task. A set of representative faces is necessary to find the implicit model.

Eigenfaces [56] were used to model the distribution of faces in some large input space (typically the input space is $\mathbb{R}^n$ where n is the number of pixels in the image). The main assumption is that the set of faces are localized in a subspace that can be approximated using a training set. The "faceness" of an input image is determined by its distance to the face subspace.

A similar approach was proposed by Sung and Poggio [54] where the subspace was approximated by Gaussian distributions. The distances between a given input image and the subspaces generated a vector that was used by an Multi-Layer Perceptron[1], to separate the face space from the non-face space.

Another approach often used is only based on MLPs [50] [17] [54] [16]. This approach uses a set of face patterns to train an MLP. This set has to be the most representative of human faces and is reduced to a specific template size (19x19 pixels or 15x20 pixels for instance). The models are trained either in a discriminant framework (predicting the class, face or not face), or in a generative framework (predicting the input and then computing a distance between the input and its reconstructed version). Then, the trained model is tested over a set of images containing various faces (location and scale) (Figure 3.7) and a large set of non-face images. The first set is used to evaluate the false rejection rate using ground truth coordinates generally provided with test images. The second set is used to evaluate the false alarm rate, i.e. the number of detected faces in non-face images. The false alarm rate has to be low regarding the number of tests in one image.

The investigations on neural networks as a tool for face detection have shown their reliability and robustness for such a task [57]. However, the time consuming processing [46, 12] needed by the neural

[1]see section 4.2 for an introduction to Multi-Layer Perceptron (MLP).

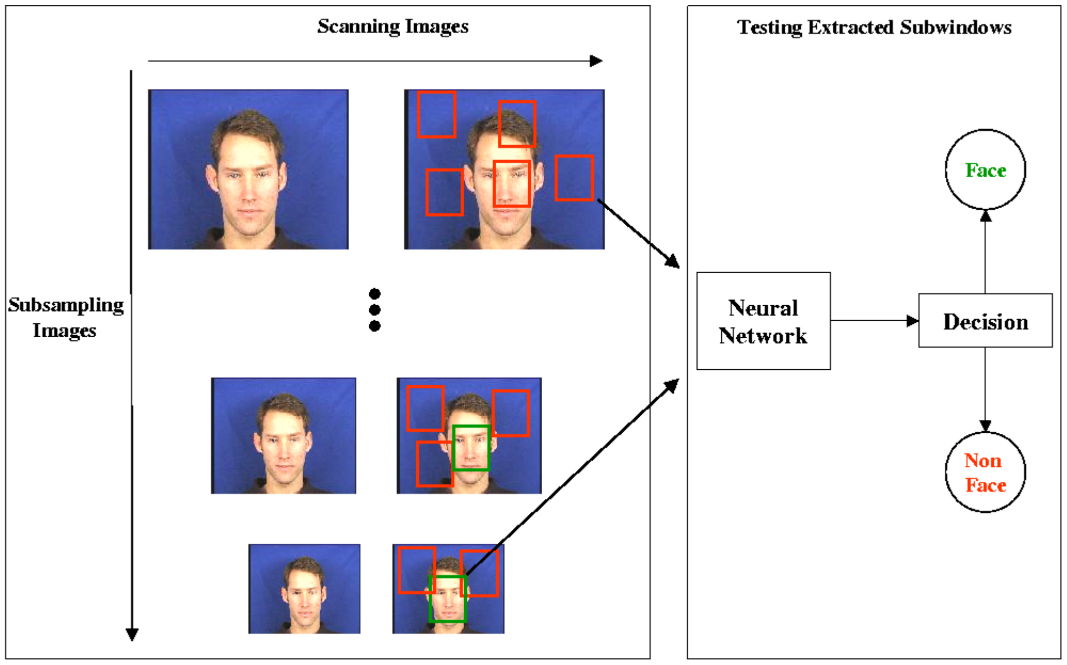


Figure 3.7: Scanning process for face detection.

networks has prevented them from being a practical tool. Most of the computation time is spent in exploring all the possible sub-images (Figure 3.7). Although some strategies were used to reduce the time complexity, this also reduced the performance of the system.

3.3 General Framework of Face Verification Systems

In this section, the general framework of a face verification system is addressed. A face verification system can be divided into two steps.

- **The training step** consists in building a representative set of parameters for each client. This set of parameters is called the model.

  For each client, a set of reference images is available. For a given client, the training step builds a model using reference images of this client and eventually reference images from other people such as others clients (Figure 3.8).

- **The decision step** consists in comparing a given image to the model, previously computed during the training step, corresponding to the claimed identity.

The aim of the decision step (Figure 3.9) is, given a unknown face image X or Y claiming an identity #, to accept or reject the face image. The face image X of an unknown person (but in reality client #), claiming identity # is compared to the model #, then the client X is accepted. The face image Y of an unknown person (but in reality an impostor), claiming identity # is compared to the model #, then the impostor Y is rejected.

3.4 State-of-the-art Face Verification Algorithms

The problem of face verification has been addressed by different researcher and with different methods. For a complete survey and comparison of different approaches see [11, 67]. In the following subsection,

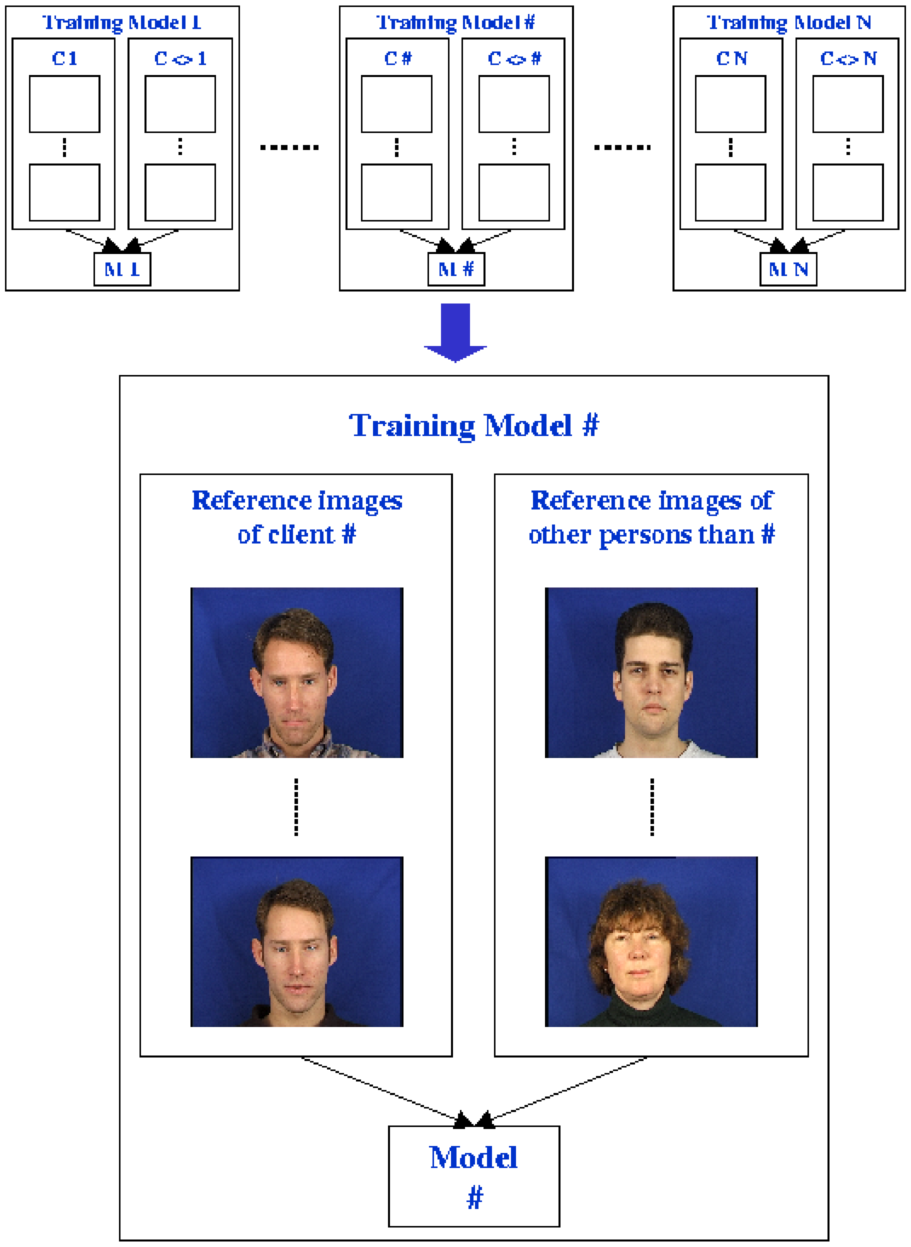


Figure 3.8: General Framework of the training step of a face verification system.

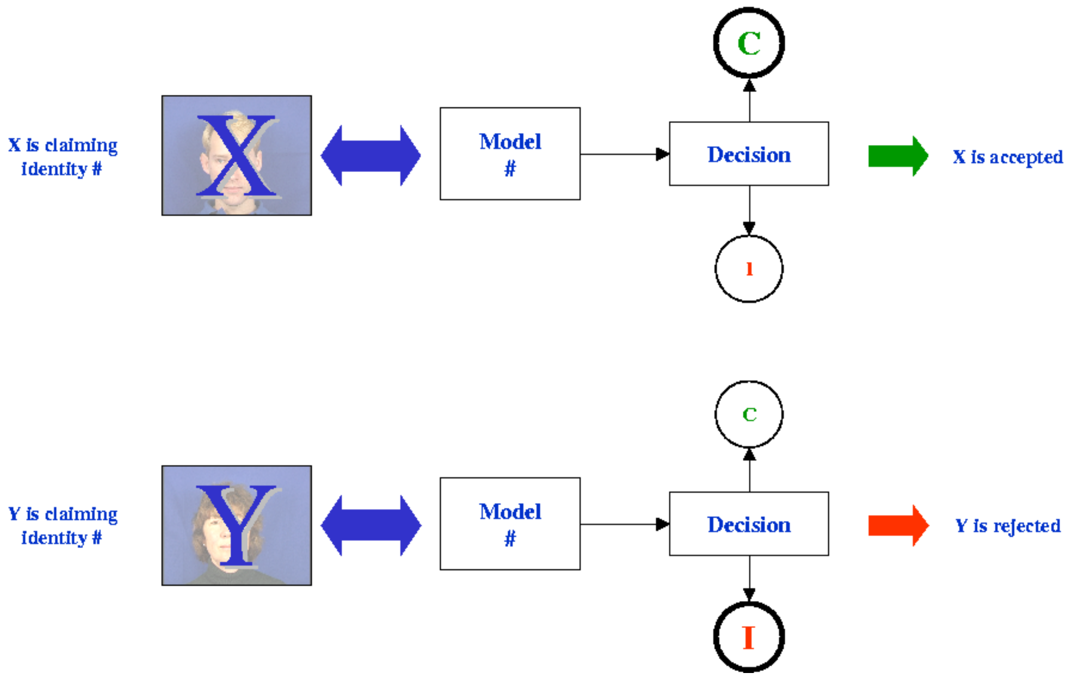


Figure 3.9: General Framework of the decision step of a face verification system.

we briefly introduce some of these methods:

3.4.1 Chamfer Matching Algorithm

This algorithm is based on a chamfer matching [42] that directly works on the profile contour encoded as x-y coordinates. The chamfer matching technique [43] searches for the best match between two binary images. Geometric transformations are used to distort one image (candidate image) to another (reference image) in order to minimize a given distance measure between them. These binary images are often derived from image edges. The algorithm generates a distance map from the reference profile. This distance map associates each pixel of the reference profile picture with its distance from the closest profile pixel. The candidate profile is projected onto the reference distance map and a global distance is computed. By minimizing the distance, the optimum compensation parameters are found (translation, rotation, scale factor). Then, the residual distance between the best compensated and the reference profiles is sent to the supervisor to decide whether the two profiles belong to the same person.

The Mean Squared Error (MSE) criterion is used to compute the distance between the two images. In order to get rid of the face variability over the different shots, the gray level distance is computed inside a rectangular window that covers the most invariant features found inside the frontal view, namely the eyes/eyebrows and nose/nostrils features. In the case of the frontal face matching, it is not possible to know a priori which parameters to apply in order to match the candidate's eye/nose window onto the reference image. The simplex algorithm is used in order to find the optimal frontal compensation parameters, and minimize the gray level distance between the two images.

3.4.2 Linear Discriminant Analysis and Support Vector Machines

In this approach, [24] adopt a client-specific solution which requires learning client-specific support vectors. Faces are represented in both Principal Component (PC) and Linear Discriminant (LD) subspaces.

The aim of the Principal Component Analysis is to identify the subspace of the image space spanned by the training face image data and to decorrelate the pixel values. This can be achieved by finding the eigenvectors of matrix associated with nonzero eigenvalues. These eigenvectors are referred to as Eigenfaces. The classical representation of a face image is obtained by projecting it to the coordinate system defined by the Eigenfaces. The projection of face images into the Principal Component (Eigenface) subspace achieves information compression, decorrelation and dimensionality reduction to facilitate decision making.

If one is also interested in identifying important attributes (features) for face verification, one can adopt a feature extraction mapping. A popular technique is to find the Fisher linear discriminant (see section 4.5).

The linear discriminant analysis (LDA) subspace holds more dicriminant features for classification than the principal component analysis (PCA) subspace. The LDA based features for personal identity verification is theorically superior to that achievable with the features computed using PCA [56] and many others [1] [14]. The projection of a face image into the system of Fisher-faces associated with nonzero eigenvalues will yield a representation which will emphasize the discriminatory content of the image.

The main decision making tool is Support Vector Machines (SVMs). The reader is referred to section 4.1 or to [10] for a comprehensive introduction of SVMs.

3.4.3 Elastic Graph Matching

The Elastic Graph Matching (EGM) [28] [18] introduces a specific face representation as illustrated in Fig. 3.10. Each face is represented by a set of feature vectors positioned on nodes of a coarse, rectangular grid placed on the image. As features the modulus of complex Gabor responses from filters with 6 orientations and 3 resolutions are used.

Comparing two faces corresponds to matching and adapting a grid taken from one image to the features of the other image. Therefore both the feature vectors of each node and the deformation information attached to the edges are taken into account. The quality of different matches between an observed grid and a reference grid can be evaluated using the following distance:

$$\begin{aligned} d(G,R) &= \sum_{i=1}^{N_n} d_n(G_{n_i}, R_{n_i}) + \lambda \sum_{j=1}^{N_e} d_e(G_{e_j}, R_{e_j}) & (3.1) \\ &= \sum_{i=1}^{N_n} d_{n_i} + \lambda \sum_{j=1}^{N_e} d_{e_j} & (3.2) \end{aligned}$$

where G_{n_i} represents the ith node of grid G, R_{e_j} is the jth node of grid R; N_n, N_e are the number of nodes and edges, respectively, and λ is a weighting factor which characterizes the stiffness of the graph. A *plastic* graph which opposes no reaction to deformation corresponds to $\lambda = 0$, while a totally rigid graph is obtained with very large values of λ.

Because of the large number of possible matches an approximate solution in [29] was proposed. The matching consists of two consecutive steps: rigid matching and deformable matching. In rigid matching an approximate match is estimated, which corresponds to setting a high value of λ. In deformable matching the grid is deformed in order to minimize (3.1). Advantages of the elastic graph matching are the robustness against variation in face position, and expression. This owes to the Gabor features, the rigid matching stage, and the deformable matching stage. If the Eigenface is used a scale and face position compensations are needed.

We note here that the contribution from nodes are considered equally. This is a drawback of the algorithm since the contributions of each node to the distance are different. Extensive experiments have been made to assess the accuracy and reliability of this method, for more details see [6].

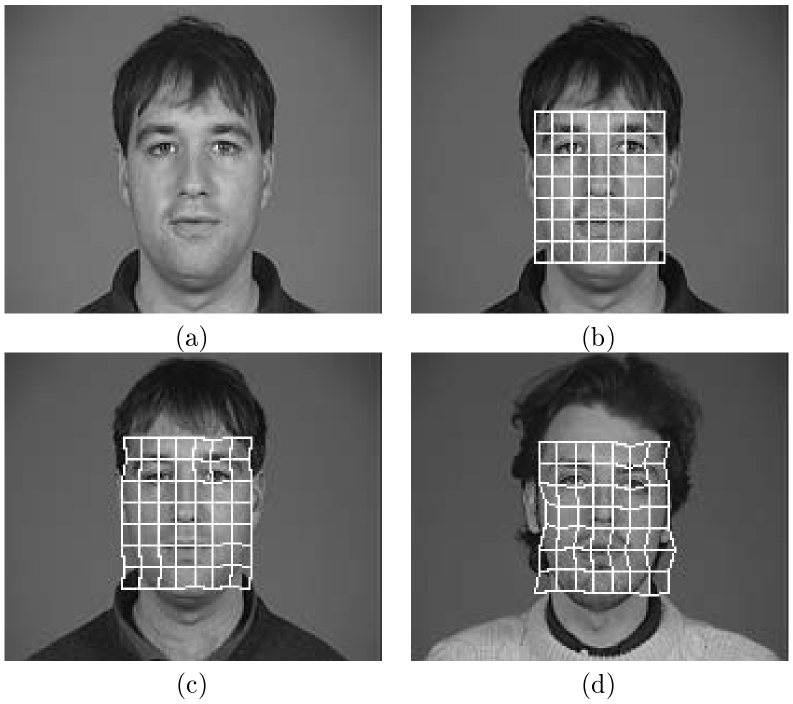


Figure 3.10: Example of a grid matching. (a) reference image, (b) reference grid, (c) matched grid on another image of the same person, (d) matched grid on another person.

3.4.4 Robust correlation

In the face verification method based on robust correlation [23], the registration is achieved by direct minimization of the robustified correlation score over a multi-dimensional search space. The search space is defined by the set of all valid geometric and photometric transformations. In the current implementation method the geometric transformations are translation, scaling and rotation.

A score function is used to evaluate a match between the transformed model image and the probe image. Parameters of the score function are purely geometrical and intensity values are not transformed. In a previous work [37], parameters are included for affine compensation of global illumination changes (gain, offset) into the search space. For efficiency reasons, a less sophisticated approach is adopted in which the histogram of residual errors is shifted (for each point in the search space) using the median error. To find the global extremum of the score function we employ a stochastic search technique incorporating gradient information.

To meet real-time requirements of the verification scenario, a multi-resolution scheme is adopt in the spatial domain. This is achieved by applying the combined gradient-based and stochastic optimization described above to each level of a Gaussian pyramid. The estimate obtained on one level is used to initialize the search at the next level. In addition to the speed-up, the multi-resolution search also has the benefit of removing local optima from the search space and thus effectively improving the convergence characteristics of the method.

In the training phase a feature selection procedure is employed based on minimizing the intra-class variance and at the same time maximizing the inter-class variance. A feature criterion is evaluated for each pixel and the subset of pixels that best discriminates a given client from other clients in the database (effectively modeling the impostor distribution) are selected. This feature subset is then used in verification allowing efficient identification of the probed image.

Chapter 4

Fusion Algorithms

Fusion algorithms are methods which goal is to merge the prediction of many algorithms in order to hope for a better average performance than any of the combined methods.

It has already been shown in many research papers that combining biometric verification systems enables to achieve better performance than techniques based on only one biometric modality [2, 9, 26, 25]. More specifically, audio-visual biometric verifications systems (based on the face and the voice of an individual) have also been extensively studied [3, 4].

Most classification machine learning algorithms can be used for fusion purposes. A good introduction to machine learning algorithms can be found in [8, 22, 59].

In [3], the authors have compared the following fusion algorithms on the XM2VTS database described in chapter 5: Support Vector Machines, Naive Bayes Classifiers, Linear Discriminant Classifiers, Decision Trees, and Multi-Layer Perceptrons. In the following, we will briefly introduce these methods. Note that for each method, we will assume that we have access to a training dataset of l pairs $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i$ is a vector containing the scores or decisions of the basic modalities (such as speaker and face verification modules), while y_i is the class of the corresponding accesses (for instance, *client* or *impostor*, often coded respectively as 1 and -1).

4.1 Support Vector Machines

Support Vector Machines (SVMs) [59] have been applied to many classification problems, generally yielding good performance compared to other algorithms. The resulting function is of the form

$$\hat{y} = \text{sign}\left(\sum_{i=1}^{l} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{4.1}$$

where $\mathbf{x}$ is the input vector of a example to test, $\hat{y}$ is the decision of the model (accept if $\hat{y}$ is positive, reject otherwise), $\mathbf{x}_i$ is the input vector of the i^{th} training example, l is the number of training examples, the α_i and b are the parameters of the model, and $K(\mathbf{x}, \mathbf{x}_i)$ is a kernel function that can have different forms, such as:

$$K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^t\ \mathbf{x}_i + 1)^d \tag{4.2}$$

which leads to a Polynomial SVM with parameter d, or

$$K(\mathbf{x}, \mathbf{x}_i) = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}_i\|^2}{\sigma^2}\right) \tag{4.3}$$

which leads to a Radial Basis Function (RBF) SVM with parameter σ. Either d or σ must be selected using methods such as cross-validation.

In order to train such SVMs, one needs to solve the following quadratic optimization problem: find the parameter vector $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_l\}$ that maximize the objective function

$$Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4.4}$$

subject to the constraints

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \tag{4.5}$$

and

$$0 \leq \alpha_i \leq C \; \forall i. \tag{4.6}$$

It is important to note that the training complexity of SVMs is quadratic on the number l of examples, which makes the use of SVMs for large datasets difficult. Note however that in the resulting solution (4.1), most α_i are equal to 0, and the examples with non-zero α_i are called *support vectors.*

4.2 Multi-Layer Perceptrons

A multi-layer perceptron (MLP) is a particular architecture of artificial neural networks [8, 22], composed of layers of non-linear but differentiable parametric functions. For instance, the output $\hat{y}$ of a 1-hidden-layer MLP can be written mathematically as follows

$$\hat{y} = b + \mathbf{w} \cdot \tanh\left(\mathbf{a} + \mathbf{x} \cdot \mathbf{V}\right) \tag{4.7}$$

where the estimated output $\hat{y}$ is a function of the input vector $\mathbf{x}$, and the parameters $\{b, \mathbf{w}, \mathbf{a}, \mathbf{V}\}$. In this notation, the non-linear function tanh() returns a vector which size is equal to the number of hidden units of the MLP, which controls its capacity and should thus be chosen carefully, by cross-validation for instance.

An MLP can be trained by gradient descent using the backpropagation algorithm [51] to optimize any derivable criterion, such as the *mean squared error* (MSE):

$$\text{MSE} = \frac{1}{l} \sum_{i=1}^{l} (y_i - \hat{y}_i)^2 \; . \tag{4.8}$$

4.3 Decision Trees

Decision Trees are a family of non-parametric learning algorithms that generate trees. In these trees, each node is either (1) a leaf that gives a decision, or (2) an internal node specifying a test on one of the input attributes of a given example. Thus, in order to classify a given input vector, and starting from the root of the tree, one simply follows the nodes that satisfy the associated tests on the input vector until a leaf is reached. The decision corresponds to the class associated with that leaf.

C4.5 is one of the most known decision tree algorithms [44], and uses information theory to derive the most discriminant attributes, which will be used as tests in the internal nodes. A pruning of the tree is also performed in order to get better generalization results.

4.4 Naive Bayes Classifier

In order to take a binary classification decision, there are two main approaches: the *discriminant* approach where one finds a solution that discriminates between the classes, and the *generative* approach, where one creates a generative model for each class and then, using the Bayes theorem, takes the decision. While most researchers agree that the former gives generally better results [59], for some applications (such as speaker verification for instance), the latter is sometimes better.

In section 2.3, we have described the generative approach used for speaker verification and explained that in order to take a decision, we had to create a generative model for the client, a generative model for the non-client (or the world), and then determine a threshold for the decision. This approach, with the corresponding equations (2.9) and (2.12), describes the general Bayesian classifier framework.

In order to simplify further the model, one can make the hypothesis that the input features are independent from each other, hence the likelihood of a client represented by its m features $\{d_1, d_2, \dots, d_m\}$ coming from the decisions of m independent systems can be approximated using the *Naive Bayes* model as follows:

$$p(d_1, d_2, \cdots, d_m) = \prod_{i=1}^{m} p(d_i)\ . \tag{4.9}$$

While in section 2.3 we proposed the use of GMMs to represent the client and world models, in the experimental part of this report, the authors of [3] have chosen to represent the marginal distributions of the modalities by Beta distributions:

$$\begin{aligned} p(d_i) &= Be(\alpha_i, \beta_i) \\ &= \frac{\Gamma(\alpha_i + \beta_i)}{\Gamma(\alpha_i)\Gamma(\beta_i)} d_i^{\alpha_i - 1}(1 - d_i)^{\beta_i - 1}, \end{aligned} \tag{4.10}$$

where Γ is the gamma function, $0 \leq d_i \leq 1$, $\alpha_i > 0$ and $\beta_i > 0$. The mean μ and the variance σ^2 of the Beta distribution are given by [5]:

$$\mu = \frac{\alpha_i}{\alpha_i + \beta_i} \tag{4.11}$$

$$\sigma^2 = \frac{\alpha_i \beta_i}{(\alpha_i + \beta_i)^2(\alpha_i + \beta_i + 1)}. \tag{4.12}$$

The parameters μ and σ of each distribution can then be estimated by maximizing its likelihood using algorithms such as EM [13]. Finally, as in section 2.3, one needs to create a client model and a world model, and then estimate a threshold in order to take a decision.

4.5 Fisher Linear Discriminant

A linear discriminant is a simple linear projection of the input vector onto an output dimension:

$$\hat{y} = b + \mathbf{w} \cdot \mathbf{x} \tag{4.13}$$

where the estimated output $\hat{y}$ is a function of the input vector $\mathbf{x}$, and the parameters $\{b, \mathbf{w}\}$.

Depending on the criterion chosen to select the optimal parameters, one could obtain a different solution. The Fisher criterion [19] aims at maximizing the ratio of between-class scatter to within-class scatter.

Given a set of l_1 points belonging to class $\mathcal{C}_1$, and l_2 points belonging to class $\mathcal{C}_2$, we can define the mean of each class as

$$\mathbf{m}_1 = \frac{1}{l_1} \sum_{k \in \mathcal{C}_1} \mathbf{x}_i$$

$$\mathbf{m}_2 = \frac{1}{l_2} \sum_{k \in \mathcal{C}_2} \mathbf{x}_i \ .$$

The scatter matrices of each class is then defined as

$$\mathbf{S}_1 = \sum_{k \in \mathcal{C}_1} (\mathbf{x}_k - \mathbf{m}_1)(\mathbf{x}_k - \mathbf{m}_1)^t$$

$$\mathbf{S}_2 = \sum_{k \in \mathcal{C}_2} (\mathbf{x}_k - \mathbf{m}_2)(\mathbf{x}_k - \mathbf{m}_2)^t \ .$$

The between-class scatter matrix is defined as

$$\mathbf{S}_b = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^t \ .$$

Fisher's criterion can then be defined as maximizing

$$J(\mathbf{w}) = \frac{\mathbf{w}^t \mathbf{S}_b \mathbf{w}}{\mathbf{w}^t (\mathbf{S}_1 + \mathbf{S}_2) \mathbf{w}} \ .$$

The functional is also known as the Rayleigh quotient, and the maximum is reached when

$$\mathbf{w} = (\mathbf{S}_1 + \mathbf{S}_2)^{-1} (\mathbf{m}_1 - \mathbf{m}_2) \ .$$

Chapter 5

Evaluation of Biometric Verification

The goal of the present document is to evaluate state-of-the-art biometric techniques of identity verification. Unfortunately, available benchmark databases for such a task are not very common. A previous european project, M2VTS, have produced such databases. We have chosen the XM2VTS database [34], using its associated experimental protocol, the *Lausanne Protocol* [33]. After the description of the database, we present in this chapter an experimental comparison of one voice and one face verification system, as well as the six fusion algorithms presented in the previous chapter.

5.1 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 5.1). 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 5.1: Face images of the same persons during different sessions and shots.

5.2 Experimental Protocol

A protocol has been defined [33] 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 5.2). 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 [33] 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 5.2. 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 5.2 for the partitions):

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

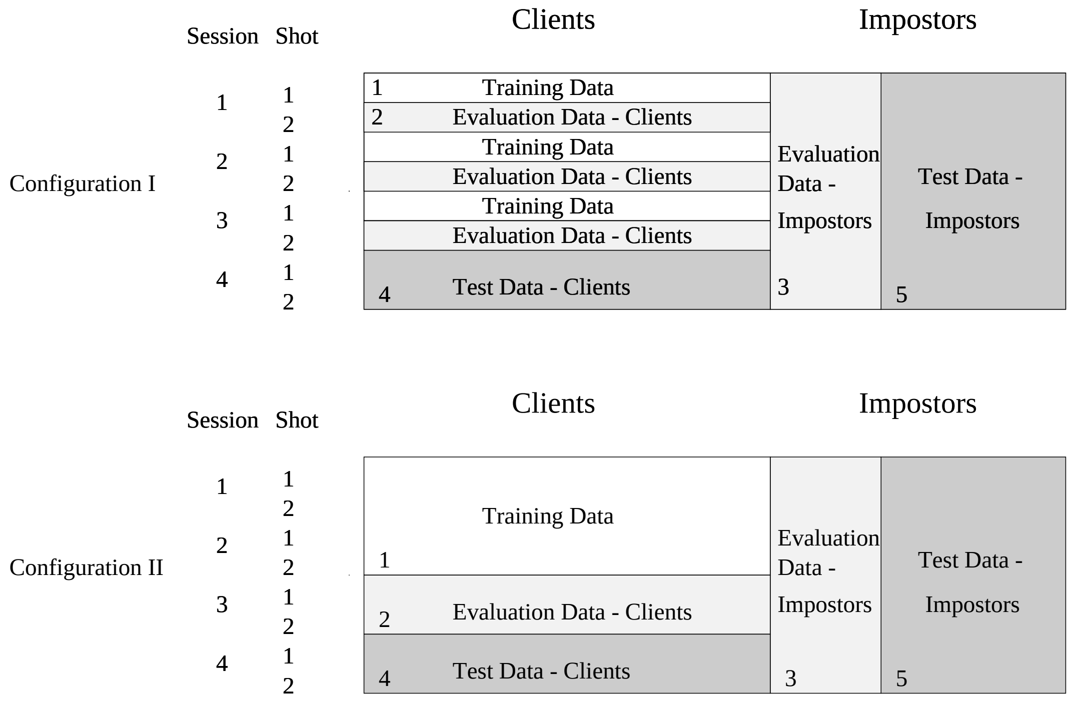


Figure 5.2: Diagramme showing the partitioning of the XM2VTS database according to the protocol Configurations I (top) and II (bottom).

5.3 Experimental Results

The results reported in this section were mainly taken from the following research papers: [3, 4, 36]. The first series of experiments report state-of-the-art results on face verification applied to the XM2VTS database. In a second series of experiments, we report a comparison of six fusion algorithms where the basic modalities where speaker and face verification systems, but unfortunately not the same as the one presented in the first table.

In Table 5.1, we show some of the results of a face verification contest [36] on XM2VTS, organized in conjunction with the International Conference on Pattern Recognition 2000. The main participants of this contest were the Aristotle University of Thessaloniki (AUT), the Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP), the University of Surrey (Surrey) and the University of Sydney (Sydney). They respectively used the following methods:

AUT: based on the Elastic Graph Matching method presented in section 3.4.3 and in [27].

IDIAP: also based on the Elastic Graph Matching method presented in section 3.4.3 and in [3].

Surrey: based on Linear Discriminant Analysis (LDA), using a novel metric and either a client-specific threshold or a global threshold, as described in section 3.4.2 and in [31].

Sydney: based on the Fractal Neighbor Distance (FND) using a generic face template and a fractal transformation, as described in [55].

All the models were selected in order to optimize the EER on the evaluation set. For each model, we show the false alarme rate (FAR), the false rejection rate (FRR), and the combined HTER obtained on the test set for the two configurations described in section 5.2.

In the second series of experiments, the face and voice verification systems were not necessarily the current state-of-the-art methods but were instead the following:

Participants	Configuration I			Configuration II		
	FAR	FRR	HTER	FAR	FRR	HTER
AUT	8.2	6.0	7.1	6.2	3.5	4.85
IDIAP	8.1	8.5	8.3	7.7	7.3	7.5
Surrey (Global Threshold)	6.5	5.3	5.9	3.5	3.8	3.65
Surrey (Client Threshold)	2.3	2.5	2.4	1.2	1.0	1.1
Sydney	13.6	12.3	12.95	13.0	12.3	12.65

Table 5.1: Performance of some state-of-the-art face verification systems on the test set of XM2VTS

Voice verification system: based on a sphericity measure and a global threshold, as described in section 2.1.2 and in [3].

Face verification system: based on the Elastic Graph Matching method and a global threshold, as described in section 3.4.3 and in [3].

These two systems were compared on the two configurations of the Lausanne protocol, together with six fusion algorithms.

In Table 5.2, we first show the results of the modalities alone (voice and face), then the results of the fusion methods described in section 4: two different SVMs, using respectively a Polynomial kernel and an RBF kernel, then the Bayes classifier, the C4.5 decision tree, the Fisher linear discriminant (LD) and finally the multilayer perceptron (MLP).

All the models were selected in order to optimize the EER on the evaluation set. For each model, we show the false alarme rate (FAR), the false rejection rate (FRR), and the combined HTER obtained on the test set for the two configurations described in section 5.2.

Modalities	Fusion Method	Configuration I			Configuration II		
		FAR	FRR	HTER	FAR	FRR	HTER
Face	-	8.08	8.50	8.29	7.76	7.25	7.51
Voice		1.60	5.00	3.30	5.53	4.25	4.89
Face and Voice	Polynomial SVMs	1.09	0.0	0.55	2.09	1.5	1.80
	RBF SVMs	1.18	0.0	0.59	3.26	1.0	2.13
	Bayes	0.63	0.0	0.32	1.5	1.75	1.63
	C4.5	0.0	3.41	1.71	0.12	4.75	2.44
	Fisher LD	1.45	0.0	0.73	69.2	0.0	34.6
	MLP	1.58	0.0	0.79	0.17	4.0	2.09

Table 5.2: Performance of different fusion classifiers on the test set

As we can see in the first part of the table, the speaker verification system has an overall performance largely better than the face verification system (but be careful as the state-of-the-art face verification systems are in fact similar to the state-of-the-art speaker verification systems). In the second part of the table, we see that most fusion method gave better results than the modalities alone, which confirms the expectations.

We are currently undertaking new fusion experiments using the most recent results from speaker and face verification systems. It should be interesting to see if fusion algorithms can still enhance the performance of already very good basic verification systems.

Chapter 6

Conclusion

In this report, we have presented an overview of state-of-the-art biometric verification algorithms, based on speaker verification as well as face verification techniques. We have explained why we should use a text independent scenario for the speaker verification algorithms. We have also presented some fusion techniques that enable one to merge the results of more than one verification method in order to obtain better results on average and sometimes confidence interval on the decisions taken.

Some experimental results have also been presented on the XM2VTS database, comparing speaker verification, face verification, as well as six fusion algorithms.

Whether these results are convincing enough to create real-life banking applications where the security is mainly based on biometric verification still remains to be established. One limitation could be the size of the underlying models (for instance GMMs perform much better than sphericity models, but have a lot more parameters). Hence one could prefer to have simpler models that can fit on some special hardware (such as smarcards), but then be ready to have poorer verification performance.

On the other hand, many open problems are still the focus of many researchers in the fields of biometric verification. For instance, in speaker verification, better preprocessing methods more adapted to speaker verification (and not to speech recognition) are also still needed, better generative models for text independent speaker verification that would take into account temporal relations, as opposed to current GMM models, are also needed, as well as understanding why the classical discriminant models such as SVMs and MLPs do not perform well on this task.

In face verification, improvements are also still needed. It could be possible to increase the efficiency of face verification algorithms by using new features. For example, state-of-the-art methods use only gray images and no color information. Currently, new methods are investigated in order to take into account the color, using skin color distribution for instance. Futhermore, the motion is not used, while the analysis of the movement of the head could be also a major point to increase performances of face verification systems. But of course, this analysis may have a high impact in terms of CPU and memory requirements, which could not be acceptable for a real-life implementation.

Finally, regarding the fusion algorithms, they already showed that they were indeed able to perform better in average than any single modality, when these modalities were not state-of-the-art systems. It will be interesting to see if fusion methods can still perform better when the basic modalities already give good performance. On the other hand, we are still interested in searching for correctly estimated confidence intervals as well as new ways of using the informations coming out of each modality model: instead of using only the score, we could for instance use some gradient information related to the score, or some information related to the specific access, such as the chanels used, or even some representation of the access itself.

Bibliography

[1] P. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. In *ECCV'96*, pages 45–58, 1996. Cambridge, United Kingdom.

[2] S. Ben-Yacoub. Multi-modal data fusion for person authentication using SVM. In *Proceedings of the Second International Conference on Audio- and Video-based Biometric Person Authentication (AVBPA'99)*, pages 25–30, 1999.

[3] S. Ben-Yacoub, Y. Abdeljaoued, and E. Mayoraz. Fusion of face and speech data for person identity verification. *IEEE Transactions on Neural Networks*, 10(05):1065–1074, 1999.

[4] S. Ben-Yacoub, J. Lüettin, K. Jonsson, J. Matas, and J. Kittler. Audio-visual person verification. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, 1999.

[5] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer, 1985.

[6] J. Bigün, B. Duc, F. Smeraldi, S. Fischer, and A. Makarov. Multi-modal person authentication. In Springer-Verlag, editor, *Face Recognition: From Theory to Applications (NATO-ASI Workshop)*, 1997.

[7] F. Bimbot. Second-order statistical measure for text-independent speaker identification. *Speech Communication*, 17(1-2):177–192, 1995.

[8] C. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[9] R. Brunelli and D. Falavigna. Person identification using multiple cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(10):955–966, 1995.

[10] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):1–47, 1998.

[11] R. Chellappa, C.L Wilson, and C.S Barnes. Human and machine recognition of faces: A survey. Technical Report CAR-TR-731, University of Maryland, 1994.

[12] M. Collobert, R. Feraud, G. Le Tourneur, and O. Bernier. Listen: A system for locating and tracking individual speakers. In *2nd International Conference on Automatic Face and Gesture Recognition*, pages 283–288, 1996.

[13] 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.

[14] Pierre A. Devijver and Josef Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, Englewood Cliffs, N.J., 1982.

[15] Ming-Hsuan Yang et Narenda Ahuja. Detecting faces in color images. In *Proceedings of the International Conference on Image Processing*, volume 1, pages 127–130, 1998.

[16] R Feraud, O. Bernier, and D. Collobert. A constrained generative model applied to face detection. *Neural Processing Letters*, 5(7381), 1997.

[17] R. Feraud, O. Bernier, and M. Collobert. A fast and accurate face detector for indexation of face images. In *4th IEEE International Conference on Automatic Face and Gesture Recognition*, 2000.

[18] S. Fischer, B. Duc, and J. Bigün. Face recognition with gabor phase and dynamic link matching for multi-modal identification. Technical Report LTS 96.04, EPFL, 1996.

[19] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(II):179–188, 1936.

[20] S. Furui. Recent advances in speaker recognition. In Springer, editor, *Audio- and Video-based Biometric Person Authentication*, pages 237–252, 1997.

[21] 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.

[22] S. Haykin. *Neural Networks, a Comprehensive Foundation, second edition.* Prentice Hall, 1999.

[23] K. Jonsson, J. Matas, and J. Kittler. Fast face localisation and verification by optimised robust correlation. Technical report, U. of Surrey, Guildford, Surrey, United Kingdom, 1997.

[24] K. Jonsson, J. Matas, J. Kittler, and Y.P. Li. Learning support vectors for face verification and recognition. In *4th International Conference on Automatic Face and Gesture Recognition*, pages 208–213, 2000.

[25] J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis and Applications*, 1:18–27, 1998.

[26] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[27] C. Koutropoulos, A. Tefas, and I. Pitas. Morphological elastic graph matching applied to frontal face authentication under well-controlled and real conditions. Technical report, Aristotle University of Thessaloniki, 1999.

[28] M. Lades, J. Buhmann, J. C. Vorbrüggen, J. Lange, C. v.d. Malsburg, R. P. Würtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on Computers*, 42(3):300–311, 1993.

[29] M. Lades, J. Vorbruggen, J. Buhmann, J. Lange, C. V. D. Malburg, and R. Wurtz. Distortion invariant object recognition in the dynamic link architecture. *IEEE Trans. Comput.*, 42:300–311, 1993.

[30] Kung-Pu Li and Jack E. Porter. Normalizations and selection of speech segments for speaker recognition scoring. In *Proceedings of the IEEE ICASSP*, pages 595–597, 1988.

[31] Y. Li, J. Kittler, and J. Matas. On matching scores of LDA-based face verification. In T. Pridmore and D. Elliman, editors, *Proceedings of the British Machine Vision Conference BMVC2000.* British Machine Vision Association, 2000.

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

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

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

[35] 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.

[36] J. Matas, M. Hamouz, K. Jonsson, J. Kittler, Y. Li, C. Kotropoulos, A. Tefas, I. Pitas, T. Tan, H. Yan, F. Smeraldi, J. Bigun, N. Capdevielle, W. Gerstner, S. Ben-Yacoub, Y. Abdeljaoued, and E. Mayoraz. Comparison of face verification results on the XM2VTS database. In A. Sanfeliu, J. J. Villanueva, M. Vanrell, R. Alqueraz, J. Crowley, and Y. Shirai, editors, *Proceedings of the 15th ICPR*, volume 4, pages 858–863. IEEE Computer Society Press, 2000.

[37] J. Matas, K. Jonsson, and J. Kittler. Fast face localisation and verification. In *BMVC'97*, pages 152–161. BMVA Press, 1997.

[38] T. Matsui and S. Furui. Robust methods of updating model and a priori threshold in speaker verification. In *Proc. of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume I, pages 97–100, 1996.

[39] H. Melin, J.W. Koolwaaij, J. Lindberg, and F. Bimbot. A comparative evaluation of variance flooring techniques in hmm-based speaker verification. In *ICSLP 1998*, pages 1903–1906, 1998.

[40] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

[41] J.-B. Pierrot, J. Lindberg, J. Koolwaaij, H.-P. Hutter, D. Genoud, M. Blomberg, and F. Bimbot. A comparison of a priori thresholds settings procedures for speaker verification in the CAVE project. In *Proc. of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume I, pages 125–128, 1998.

[42] S. Pigeon and L. Vandendorpe. Profile authentification using a chamfer matching algorithm. In *AVBPA'97*, pages 185–192, 1997. Crans-Montana, Switzerland.

[43] S. Pigeon and L. Vandendorpe. Image-based multimodal face authentification. *Signal Processing*, 69(1):59–79, 1998.

[44] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

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

[46] M.J.T Reinders, R.W.C Koch, and J.J Gerbrands. Locating facial features in image sequences using neural networks. In 2^{nd} *International Conference on Automatic Face and Gesture Recognition*, pages 230–235, 1996.

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

[48] P. Richard, Schumeyer, and Kenneth E. Barner. A Color-Based Classifier for region identification in video. In *Proceedings of Visual Communications and Image Procesing*, pages 189–200, 1998.

[49] 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.

[50] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *Transactions on Pattern Analysis and Machine Intelligence*, 20(1), 1998.

[51] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and James L. McClelland, editors, *Parallel Distributed Processing*, volume 1. MIT Press, Cambridge, MA., 1986.

[52] Ronald W. Schafer and Lauwrence R. Rabiner. Digital representations of speech signals. In *IEEE Proc.*, volume 63, pages 662–667, April 1975.

[53] F. K. Soong, A. E. Rosenberg, L. R. Rabiner, and B.-H. Juang. A vector quantization approach to speaker recognition. In *Proceedings of the IEEE ICASSP*, pages 387–390, 1985.

[54] Kah-Kay Sung and Tomaso Poggio. Example-based learning for view-based human face detection. In *Image Understanding Workshop*, 1994.

[55] T. Tan and H. Yan. Face recognition by fractal transformations. In *Proceedings of the IEEE ICASSP*, pages 3537–3540, 1999.

[56] M. Turk and A. Pentland. Eigenface for recognition. *Journal of Cognitive Neuro-science*, 3(1):70–86, 1991.

[57] D. Valentin, H. Abdi, A.J Otoole, and G.W Cottrell. Connectionist models of face processing: A survey. *Pattern Recognition*, 27:1209–1230, 1994.

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

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

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

[61] T. Vintsyuk. Speech discrimination by dynamic programming. *Kibernetika*, 4:81–88, 1968.

[62] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, pages 260–269, 1967.

[63] J. Yang and A. Waibel. A real-time face tracker. In *3rd IEEE Workshop on Application of Computer Vision*, pages 142–147, 1996.

[64] Jie Yang, Weier Lu, and Alex Waibel. Skin color modeling and adaptation. In *Proceedings of the 3^{rd} Asian Conference on Computer Vision*, volume 2, pages 687–694, 1998.

[65] M.-H. Yang, N. Ahuja, and D. Kriegman. A survey on face detection methods. *Transactions on Pattern Analysis and Machine Intelligence*, 1999. available at http://vision.ai.uiuc.edu/mhyang/papers/survey.ps.gz.

[66] Ming-Hsuan Yang and Narenda Ahuja. Gaussian Mixture Model for Human Skin Color and Its Applications in Image and Video Databases. In *Conference on Storage and Retrieval for Image and Video Databases*, volume 3656, pages 458–466, 1999.

[67] J. Zhang, Y. Yan, and M. Lades. Face recognition: Eigenfaces, elastic matching, and neural nets. In *Proceedings of IEEE*, volume 85, pages 1422–1435, 1997.