Confidence Measures for Multimodal Identity Verification ⋆

Samy Bengio ∗, Christine Marcel, Sébastien Marcel, Johnny Mariéthoz

Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP), CP 592, rue du Simplon 4, 1920 Martigny, Switzerland

Abstract

Multimodal fusion for identity verification has already shown great improvement compared to unimodal algorithms. In this paper, we propose to integrate confidence measures during the fusion process. We present a comparison of three different methods to generate such confidence information from unimodal identity verification systems. These methods can be used either to enhance the performance of a multimodal fusion algorithm or to obtain a confidence level on the decisions taken by the system. All the algorithms are compared on the same benchmark database, namely XM2VTS, containing both speech and face information. Results show that some confidence measures did improve statistically significantly the performance, while other measures produced reliable confidence levels over the fusion decisions.

Key words: confidence level, multimodal fusion, identity verification, machine learning algorithms, speaker verification, face verification, SVM, MLP, Bayes Classifiers, GMM, Fisher score

1 Introduction

Identity verification is a general task that has many real-life applications such as access control or transaction authentication. Biometric identity verification

⋆ This research has been partially carried out in the framework of the European BANCA project, funded by the Swiss OFES project number 99-0563-1.

∗ Corresponding author.

Email addresses: `bengio@idiap.ch` (Samy Bengio), `Christine.Marcel@idiap.ch` (Christine Marcel), `marcel@idiap.ch` (Sébastien Marcel), `marietho@idiap.ch` (Johnny Mariéthoz).

systems are based on the characteristics of a person, such as face, fingerprint or signature [1]. While a lot of research is devoted to the development of unimodal biometric verification systems, using for instance the voice or the face image of a person (see for instance [2–6]), recent papers have shown that combining such unimodal verification systems can enhance the overall performance of the system (see for instance [7–13,4,14,15]). The combination is sometimes performed using statistical assumptions about the unimodal scores (such as gaussianity, bounded score domain, or simply the fact that the scores can be considered as probabilities) or using machine learning algorithms such as Multi-Layer Perceptrons or Support Vector Machines [16–18].

While these fusion algorithms use either the decisions (accept or reject the access) or the scores obtained by the unimodal algorithms in order to take a global decision, we propose in this paper to provide other complementary information to the fusion algorithms, based on an estimation of the confidence one has on the unimodal algorithm scores. This paper thus proposes three different methods to estimate such confidence on the scores. The first two methods are based on the scores obtained by the unimodal verification systems, while the last one is based on the average gradient amplitude of the score of the unimodal verification systems, with respect to all the parameters. This last method, combined with either Support Vector Machines or Multi-Layer Perceptrons, yields significantly better performance than to the corresponding performance without confidence measure.

On the other hand, for some identity verification applications, it might be desirable not to take a decision when the score given by the verification system (either unimodal or multimodal) is uncertain. Confidence information could then be used to select a threshold under which no decision is taken (the decision is then delayed or put into the hands of a human decider). The second method proposed in this paper, which estimates the confidence on the decision using a non parametric algorithm, has been used for this purpose and results show that overall performances are indeed enhanced when only a small fraction of the accesses are referred to a human decider.

In the next section we first introduce the reader to the problem of identity verification, based either on voice (hence, speaker verification) or face image (face verification). Afterward, in section 3 we explain how fusion algorithms could be used to enhance the decision of unimodal identity verification systems. In section 4, we propose three different methods that estimate a confidence measure based either on the decisions of the unimodal systems or on their structure. In section 5, we propose an experimental comparison of the use of these confidence measures to enhance the quality of the decisions taken by the fusion algorithms or simply to provide a confidence measure on the decisions taken. We compare three different confidence measures, as well as three different fusion algorithms, on the well-known benchmark database XM2VTS using its associated Lausanne protocol. Finally, we analyze the results and conclude.

2 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 [1].

An identity verification system has 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:

$$\mathrm{FAR} = \frac{\text{number of FAs}}{\text{number of impostor accesses}}, \tag{1}$$

$$\mathrm{FRR} = \frac{\text{number of FRs}}{\text{number of client accesses}}. \tag{2}$$

A unique measure often used [19] (particularly during NIST evaluations) combines these two ratios into the so-called *decision cost function* (DCF) as follows:

$$\mathrm{DCF} = \mathrm{Cost}(\mathrm{FR}) \cdot P(\text{client}) \cdot \mathrm{FRR} + \mathrm{Cost}(\mathrm{FA}) \cdot P(\text{impostor}) \cdot \mathrm{FAR} \tag{3}$$

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, $\mathrm{Cost}(\mathrm{FR})$ is the cost of a false rejection, and $\mathrm{Cost}(\mathrm{FA})$ 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:

$$\text{HTER} = \frac{\text{FAR} + \text{FRR}}{2} . \tag{4}$$

Most verification systems output a score for each access. Selecting a threshold over which scores are considered genuine clients instead of impostors can greatly modify the relative performance of FAR and FRR. A typical threshold chosen is the one that reaches the *Equal Error Rate* (EER) where FAR=FRR on a separate validation set [20]. Note that EER and HTER, while similar, are different concepts: EER is often used to select a threshold but cannot be used to measure the performance of a system on unknown data, while HTER can be used to measure this performance.

Another method to evaluate the performance of a system is through the use of the so-called *Receiver Operating Characteristic* (ROC) curve, which represents the FAR as a function of the FRR. A more interesting version of the plot is the DET curve, which is a non-linear transformation of the ROC curve in order to make results easier to compare [21]. 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 4 shows examples of DET curves.

In the following subsections, we briefly introduce the two unimodal identity verification systems used in the present study.

2.1 Baseline Speaker Verification System

Classical speaker verification systems are based on statistical 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:

$$P(S_i|\mathbf{X}) = \frac{p(\mathbf{X}|S_i)P(S_i)}{p(\mathbf{X})}. \tag{5}$$

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 $P(\bar{S}_i|\mathbf{X})$. When $P(\bar{S}_i|\mathbf{X})$ is the same for all clients S_i, which is the case in this paper, we replace it by a speaker independent model $P(\Omega|\mathbf{X})$ where Ω represents the *world* of all the speakers. The decision rule is then:

$$\text{if } P(S_i|\mathbf{X}) > P(\Omega|\mathbf{X}) \text{ then } \mathbf{X} \text{ was generated by } S_i. \tag{6}$$

Using equation (5), inequality (6) can then be rewritten as:

$$\frac{p(\mathbf{X}|S_i)}{p(\mathbf{X}|\Omega)} > \frac{P(\Omega)}{P(S_i)} = \delta_i \tag{7}$$

where the ratio of the prior probabilities is usually replaced by a threshold δ_i since it does not depend on $\mathbf{X}$. Taking the logarithm of (7) leads to the *log likelihood ratio*:

$$\log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\Omega) > \log \delta_i = \Delta_i. \tag{8}$$

To implement this, we 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. In fact, this threshold is often forced to be the same for each client due to the lack of data to estimate it adequately. In this paper, the decision function (and thus the threshold) is replaced by the fusion process described in section 3.

The most used model, in the context of text-independent speaker verification, is the Gaussian Mixture Model (GMM) with diagonal covariance matrix. In order to use such a model, we make the (often false) assumptions that the frames of the speech utterance are independent from each other and the features in each frame are uncorrelated: the probability of a sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ given a GMM with N Gaussians is computed as follows

$$p(\mathbf{X}) = \prod_{t=1}^{T} p(\mathbf{x}_t) = \prod_{t=1}^{T} \sum_{n=1}^{N} w_n \cdot \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\sigma}_n) \tag{9}$$

where w_n is the weight of Gaussian $\mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\sigma}_n)$ with mean $\boldsymbol{\mu}_n \in \mathbb{R}^d$ where d is the number of features and with standard deviation $\boldsymbol{\sigma}_n \in \mathbb{R}^d$. GMMs are usually trained with the EM algorithm [22] in order to maximize the likelihood of the data. Moreover, as the number of client accesses if often too small to train adequately client GMMs from scratch, adaptation methods are often used [23], taking the world model parameters as prior over the client parameters.

In the experiments reported in this paper, the sentences X were preprocessed using 16 *Mel Frequency Cepstrum Coefficients* (MFCC) [24] as well as their first derivative; moreover the client models were trained using the Bayesian MAP adaptation approach [23], starting from the world model. The overall method has also been described in more details in [25].

2.2 Baseline Face Verification System

The classical face verification process can be decomposed into several steps, namely *image acquisition* (grab the images, from a camera or a VCR, in color or gray levels), *image processing* (apply filtering algorithms in order to enhance important features and to reduce the noise), *face detection* (detect and localize an eventual face in a given image) and finally *face verification* itself, which consists in verifying if the given face corresponds to the claimed identity of the client.

In this paper, we assume (as it is often done in comparable studies, but nonetheless incorrectly) that the first three steps have been performed perfectly and we thus concentrate on the last step, namely the face verification step. A good survey on the different methods used in face verification can be found in [2,3].

Our face verification method (alse described in [4]) is based on Multi-Layer Perceptrons (MLPs) [16,17]. For each client, an MLP is trained to classify an image to be either the given client or not. The input of the MLP is a feature vector with 396 dimensions corresponding to the downsized gray level face image and to the RGB (Red-Green-Blue) skin color distribution of the face (Figure 1): first, the located face is cropped, downsized to a 15x20 image and enhanced using standard image processing such as histogram equalization and smoothing by convolving the image by a 3x3 Gaussian kernel [1]; second, skin color pixels are filtered, from the sub-image corresponding to the located face, using a fast look-up indexing table of skin color pixels; then, for each color channel, a histogram is built using 32 samples.

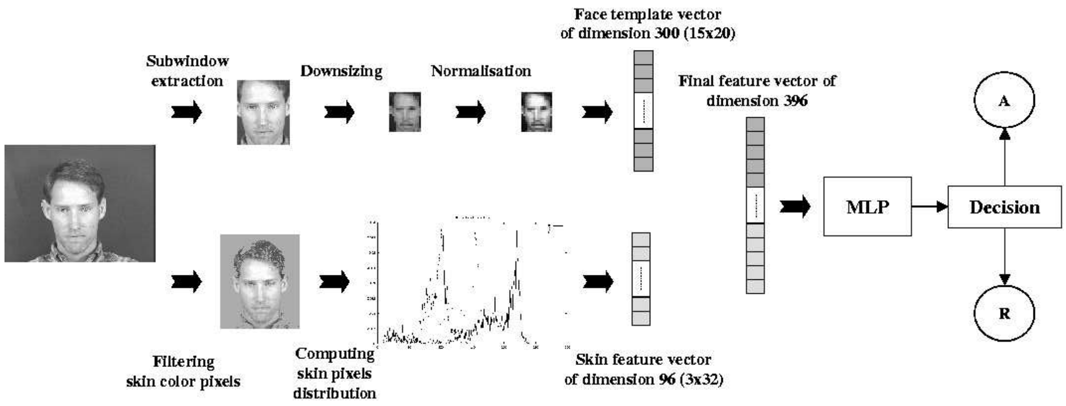

Fig. 1. An MLP for face verification using the image of the face and the distribution of the skin color.

The output of the MLP is either 1 (if the input corresponds to a client) or -1 (if the input corresponds to an impostor). The MLP is trained using both client images and impostor images, often taken to be the images corresponding to other available clients (in the present study, we used the other 199 clients of the XM2VTS database, described in section 5.1).

Finally, the decision to accept or reject a client access depends on the score obtained by the corresponding MLP which could be either above (accept) or under (reject) a given threshold, chosen on a separate validation set to optimize a criterion such as the EER.

[1] The standard deviation of the Gaussian was set to 0.5.

3 Fusion Algorithms

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

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 [7–10]. More specifically, audio-visual biometric verification systems (based on the face and the voice of an individual) have also been extensively studied [11,12].

Most classification machine learning algorithms can be used for fusion purposes. A good introduction to machine learning algorithms can be found in [16–18]. In this study, we selected the following fusion algorithms: Multi-Layer Perceptrons (MLPs) [16,17], Support Vector Machines (SVMs) [18], and Bayes Classifiers using Gaussian Mixture Models (GMMs) as density estimators [16] [2].

For each of these methods, we assume that we have access to a training dataset of 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) as well as eventual confidence informations over these scores, while y_i is the class of the corresponding accesses (for instance, *client* or *impostor*, often coded respectively as 1 and -1).

MLPs are trained to minimize the mean squared error between the obtained output and the desired output (-1 or 1); SVMs are trained to maximize the minimum margin between the correct decision and the no-decision hyperplane; Finally, Bayes Classifiers are trained by separately maximizing the likelihood of the client accesses on one GMM and the impostor accesses on a second GMM.

Each of these methods require to select one or more *hyper-parameters* in order to give its optimal performance. These hyper-parameters often control the *capacity* [18] of the model, which is related to the size of the function space spanned by the model. These hyper-parameters should be selected according to the number of training examples and the complexity of the training problem. For MLPs, this amounts to selecting the number of hidden units and/or the number of training iterations; for SVMs, it means selecting the parameter of the kernel chosen, such as the variance of the Gaussian kernel; for Bayes Classifiers using GMMs, it means selecting the number of Gaussians in the

[2] In this paper, we will assume that the reader is familiar with these models. More information could be found in the already cited references [16–18].

mixtures corresponding respectively to client and impostor models.

As there is only one training set available in general, the same data should be used to select the hyper-parameters of the models, to train them, and finally to select a decision threshold that optimizes a given criterion (in this paper, we used the EER). In order to do all this with the same dataset without the risk of optimistic bias, a cross-validation method had to be used. The method used in this paper was the following:

(1) For each value of the hyper-parameter of the fusion model,
  (a) perform a K-fold cross-validation in order to obtain unbiased scores for each train examples (in our case, $K = 5$, hence we train a model with the first $\frac{4}{5}$ of the dataset, and save the scores of the last $\frac{1}{5}$, then we do the same for each of the 4 other partitions $\{\frac{4}{5}, \frac{1}{5}\}$, in order to finally obtain scores for the whole dataset that were taken by a model that was not trained on the corresponding examples);
  (b) after a random shuffling of the data, perform a K-fold cross-validation ($K = 5$) on the obtained scores in order to compute for each partition the HTER corresponding to the threshold that optimized the EER on the other partitions;
  (c) the performance of the hyper-parameter corresponds to the average of these HTER, which are unbiased.
(2) Select the value of the hyper-parameter that has the best average HTER performance.
(3) Using the unbiased scores corresponding to the best model, select the threshold that optimizes EER on the whole training set.
(4) Train the best model over the whole training set.
(5) Apply the best model (found is step 4) on the test set and use the best threshold (found in step 3) to take the decisions.

Note that in the experiments described later in this paper, using the XM2VTS database, the training set used to train the fusion model and select the threshold was called the *evaluation set*.

As the database used for the experiments is highly unbalanced (the number of impostor accesses is more than one hundred times higher than the number of client accesses), special care should be taken during cross-validation in order to maintain the original class distribution in each fold of the K-fold procedure. Moreover, as some of the fusion algorithms strongly depend on random initialization (such as MLPs), it is advisable to run each training experiment more than once during the selection and the estimation process. In this study, we set the repeat factor to 5 during the selection process and all the results reported are averaged over 10 experiments.

4 Confidence Estimation

One can think of the fusion algorithms as a way to somehow *weight* the scores of different unimodal verification systems, eventually in a nonlinear way, in order to give a better estimation of the overall score. If one had access not only to the scores but also to a confidence measure on these scores, this measure could probably help in the fusion process. Indeed, if for some reason one unimodal verification system was able to say that its score for a given access was not very precise and should not be taken for granted, while a second unimodal verification system was more confident on its own score, the fusion algorithm should be able to provide a better decision than without this knowledge.

Hence, in this section, we propose three methods that can be used to estimate a measure of confidence over a score. In the experimental section, these three methods will be compared to see if they are indeed providing useful information that enhances the overall performance of the fusion algorithms.

Moreover, these methods could also be used to measure the confidence of the fusion algorithm decisions themselves. For some applications, it might be important to take decisions only when a high level of confidence is provided. At least one of the methods presented in this section could be used for this purpose.

4.1 Gaussian Hypothesis

One of the simplest method to estimate a confidence over a score is to do a Gaussian hypothesis on the score distribution. More specifically, suppose that all the scores s from genuine clients have been generated by the same Gaussian distribution $\mathcal{N}(s; \mu_c, \sigma_c)$ and all the scores s from impostors have been generated by another Gaussian distribution $\mathcal{N}(s; \mu_i, \sigma_i)$. Then, a good measure of confidence for a given score could be related to the distance between the probability that the score is from a client and the probability that the score is from an impostor. For instance, one could simply compute the following measure m_{gauss}:

$$m_{gauss}(s) = |\mathcal{N}(s; \mu_c, \sigma_c) - \mathcal{N}(s; \mu_i, \sigma_i)| \ . \tag{10}$$

In order to estimate the parameters $\{\mu_c, \sigma_c, \mu_i, \sigma_i\}$ we simply use a training set of scores with their associated tag (client or impostor).

The first step, however, should be to verify whether the client scores and the impostor scores could have indeed been generated by two separate Gaussians.

In order to verify whether a given set of data have been generated by a given distribution, one can use one of the many statistical tests available. In this study, we used the Kolmogorov-Smirnov statistics ks:

$$ks = \sqrt{l} \sup_s |\mathcal{N}(s; \mu, \sigma) - \hat{p}(s)| \tag{11}$$

where l is the number of examples, $\hat{p}(s)$ is the empirical distribution of the scores and $\mathcal{N}(s; \mu, \sigma)$ is the hypothesized Gaussian. Under the null hypothesis of Gaussian data, the asymptotic distribution of ks [26] is

$$\lim_{l \to \infty} p(ks < \epsilon) = 1 - 2 \sum_{k=1}^{\infty} (-1)^{k-1} \exp(-2\epsilon^2 k^2). \tag{12}$$

The sum can be closely approximated with its first few terms, because of its exponential convergence.

Unfortunately but as expected, the distribution of scores coming from the modalities do not appear to be always Gaussian. In Figure 2, we show the cumulative distribution of the scores obtained by clients from configuration I of the XM2VTS database (described in section 5.1) using the face verification modality as well as the nearest potential Gaussian cumulative distribution. Indeed, the Kolmogorov-Smirnov statistics gives a 0% probability of the data to be Gaussian.

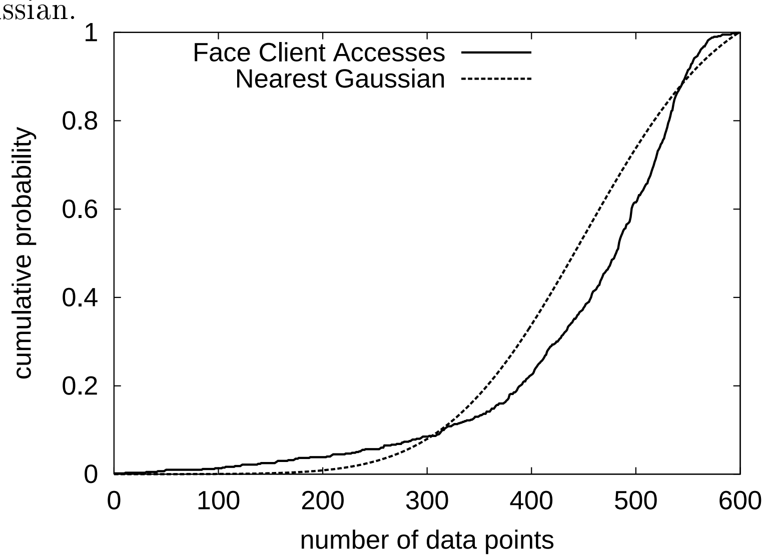

Fig. 2. Cumulative distribution of the face verification client accesses (XM2VTS database, configuration I, evaluation set) compared to the nearest Gaussian cumulative distribution.

Although the distribution of scores appear not to be Gaussian, it can still be interesting to see how a model based on this false hypothesis perform. After all, even though the distribution is not Gaussian, it can still be often close to it. In fact, it is often the case that in empirical studies, one assumes hypotheses which turn out to be false, even though the results on actual data remain good.

4.2 Non Parametric Estimation

In the (probable) event of non Gaussian scores, one can try to estimate the confidence using a non parametric model. In the following we propose a simple idea that makes no hypothesis on the distribution of the scores.

Using a training set of scores and their associated tag (client or impostor), partition the space of the scores into K distinct subspaces. The score space is unidimensional (it generally spans the real domain or a bounded version of it), so this partition is not subject to the curse of dimensionality. The partition process could be uniform over the score space (each partition has the same size) or over the score distribution (each partition contains the same number of training scores; in the experiments reported here, we have chosen this method). In both cases, simply compute for each subspace SS_i the statistics of interest. In our case, we computed m_{np}, the number of errors that were made in the subspace (false acceptances and false rejections), divided by the total number of scores in the subspace:

$$m_{np}(SS_i) = \frac{\text{number of FAs } + \text{number of FRs in } SS_i}{\text{number of accesses in } SS_i} \,. \tag{13}$$

This number gives indeed a simple confidence on the quality of the scores in SS_i. Turning to the test set, when one wants to compute the confidence of a given score s with unknown tag (client or impostor), one simply finds the subspace SS_i corresponding to the given score and returns the associated value $m_{np}(SS_i(s))$.

One question remains: how to select K. In fact, when K is small, each partition contains a large number of scores and the statistics in each partition is thus well estimated; however during the test step, the granularity of the confidence levels will be small (there will never be more than K different values of the statistics). On the contrary, when K is large, the granularity is bigger, but the estimation of the statistics in each partition is less reliable. In the experiments presented in this paper, we have chosen a value of K somewhere in the middle, with $K = 1000$ which, given the size of the test sets (more than 100000), let more the 100 values in each partition to estimate the statistics. Note that we have tried several values of K and the overall results did not really fluctuate, hence it seems to be a robust method.

Finally, note that this non parametric method is also similar to the well-known Parzen density estimator or the K-nearest-neighbor algorithm (see for instance [16] for a description of these algorithms).

Taking into account that most unimodal verification systems (at least those we are indeed using) are based on some kind of gradient method optimizing a given criterion (for instance, GMMs used in speaker verification are trained to maximize the likelihood and MLPs used in face verification are trained to minimize the mean squared error), we propose to compute the gradient of a simple measure of confidence of the decision of the model given an access with respect to every parameter in the model. The average amplitude of such gradient gives an idea of the *adequacy* of the parameters to explain the confidence of the model on the access. Hence, a global measure m_a could be

$$m_{ma}(\mathbf{X}) = \frac{1}{M}\sum_{i=1}^{M}\left|\frac{\partial f(\mathbf{X})}{\partial \theta_i}\right| \tag{14}$$

where θ_i is one the M parameters of the model and $f(\mathbf{X})$ is a simple measure of confidence of the model given access $\mathbf{X}$.

For our speaker verification system based on GMMs, we chose

$$m_{ma}(\mathbf{X}) = \frac{1}{M_c}\sum_{i=1}^{M_c}\left|\frac{\partial \log p(\mathbf{X}|S_c)}{\partial \theta_i}\right| + \frac{1}{M_w}\sum_{i=1}^{M_w}\left|\frac{\partial \log p(\mathbf{X}|\Omega)}{\partial \theta_i}\right| \tag{15}$$

where M_c is the number of parameters in the claimed client model S_c and M_w is the number of parameters in the world model Ω. Note that in this case, the measure is strongly related to the average value of the *Fisher score* [27], which is a sufficient statistics of the generative model.

For our face verification system based on MLPs, the best measure we found was

$$m_{ma}(\mathbf{X}) = \frac{1}{M}\sum_{i=1}^{M}\left|\frac{\partial MLP(\mathbf{X})^2}{\partial \theta_i}\right| \tag{16}$$

where M is the number of parameters of the model and $MLP(\mathbf{X})$ is the output obtained on access $\mathbf{X}$. Considering the fact that the targets of the MLP where -1 and 1, the higher the absolute value of the score was, the more confident the system was, and we thus measured the overall influence of the parameters to obtain high scores (and confidence).

5 Experimental Evaluation

In this section, we present an experimental comparison between different fusion algorithms, with and without confidence information, in order to assess the quality of such information. This comparison has been done using the multimodal XM2VTS database [28], using its associated experimental protocol, the *Lausanne Protocol* [29].

5.1 Database and Protocol

The XM2VTS database contains synchronized image and speech data recorded on 295 subjects during four sessions taken at one month intervals. On each session, two recordings were made, each consisting of a speech shot and a head rotation shot.

The database was divided into three sets: a training set, an evaluation set, and a test set. 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 295 subjects were divided into a set of 200 clients, 25 evaluation impostors, and 70 test impostors. Two different evaluation configurations were defined. They differ in the distribution of client training and client evaluation data. Both the training client and evaluation client data were drawn from the same recording sessions for Configuration I which might lead to biased estimation on the evaluation set and hence poor performance on the test set. For Configuration II on the other hand, the evaluation client and test client sets are drawn from different recording sessions which might lead to more realistic results. This led to the following statistics:

- training client accesses: Conf. I: 600 (200 x 3 per client), Conf. II: 400 (200 x 2 per client)
- evaluation client accesses: Conf. I: 600 (200 x 3 per client), Conf. II: 400 (200 x 2 per client)
- evaluation impostor accesses: 40'000 (25 * 8 * 200)
- test client accesses: 400 (200 * 2)
- test impostor accesses: 112'000 (70 * 8 * 200)

Finally, note that in order to train a world model for the speaker verification system, we used additional data from a separate speech database, POLYCOST [30], which was already used in previous published experiments on XM2VTS.

5.2 Baseline Results

In this section, we present two different baseline results. First in Table 1, we show the performance of each modality alone, namely speaker verification (Voice) and face verification (Face). For both, we give results for configurations I and II, and for each configuration, FAR represents the *false acceptance rate*, FRR is the *false rejection rate*, while HTER is the *half total error rate*. The results of this table as well as the following are expressed in percentage for a better readability (hence, 3.25 is in fact 3.25% or 0.0325).

Note that the performance of the speaker verification system was statistically significantly better than the face verification system (with more than 99% confidence [3]) for both configurations. Note also that these results are competitive with recent results (using global thresholds) published on the same database (see for instance [5,11] where the best face HTER was 5.9 on configuration I and 3.65 on configuration II, while the best voice HTER was 3.3 on configuration I and 4.89 on configuration II) [4].

Modalities	Configuration I			Configuration II		
	FAR	FRR	HTER	FAR	FRR	HTER
Face	2.70	3.75	3.22	1.98	3.25	2.61
Voice	2.31	1.50	1.91	2.00	1.50	1.75

Table 1
Performance on the test set of different unimodal verification systems

Then in Table 2 we present the results of three different fusion algorithms, namely SVMs using Gaussian kernel, MLPs and Bayes Classifiers using GMMs. This table should convince the reader that fusion is a very important process since all three methods performed statistically significantly better than each modality alone, for both configurations (with more than 99% confidence). On the other hand, all three methods gave similar performance with a slight over-

[3] The statistical test used here and further in the paper to verify if the difference between two performances was significant was the following: as the number of samples used to produce the performances was high (more than 100000), we assumed that the HTER performance, which can be seen as a weighted sum of two kinds of errors (false acceptances and false rejections), can be seen as a random variable which follows a normal distribution. The standard deviation of this random variable can then be estimated by taking into account the relative counts of clients and impostors, and a t test with the Welch-Satterthwaite approximation is then performed.
[4] Note that, using a special combination algorithm, ECOC [6], normally designed for robust multiclass classification tasks, researchers were able to obtain HTER as low as 0.80 on the face verification task using configuration I; no comparable results were published for configuration II.

all advantage to SVMs. Note also that these results are better, at least for configuration II, than those published in [11] (where the best HTER obtained was 1.63).

Fusion Method	Configuration I			Configuration II		
	FAR	FRR	HTER	FAR	FRR	HTER
SVMs	0.63	0.75	0.69	0.09	0.50	0.30
MLPs	0.61	1.13	0.87	0.20	0.35	0.27
Bayes	0.66	0.78	0.72	0.15	0.45	0.30

Table 2
Average performance on the test set of different fusion classifiers

5.3 Adding Confidence to Enhance Decisions

In this section, we present a comparative study between the three confidence estimation methods presented in section 4 when the estimates are computed for each modality alone, then added to the input vector of the fusion algorithms. Table 3 shows the various results, for each confidence estimation method and each fusion algorithm, on both configurations.

Confidence Method	Fusion Method	Configuration I			Configuration II		
		FAR	FRR	HTER	FAR	FRR	HTER
Gaussian Hypothesis Method	SVMs	0.62	0.75	0.69	0.10	0.50	0.30
	MLPs	0.67	0.90	0.78	0.20	0.33	0.26
	Bayes	0.69	1.30	0.99	0.24	0.53	0.38
Non Parametric Method	SVMs	0.49	1.08	0.78	0.10	0.53	0.31
	MLPs	0.58	1.05	0.81	0.29	0.40	0.35
	Bayes	0.87	0.75	0.81	0.28	0.33	0.35
Model Adequacy Method	SVMs	0.60	0.75	0.67	0.27	0.25	0.26
	MLPs	0.49	1.00	0.79	0.18	0.25	0.22
	Bayes	0.78	1.25	0.99	0.28	0.38	0.34

Table 3
Average performance on the test set of different fusion classifiers using various confidence measure estimations

It appears that using the Bayes fusion method, none of the confidence method was able to enhance the performance. On the other hand, using either MLPs or SVMs, the Model Adequacy method was systematically better than the corresponding system without confidence information. The two other confidence methods did not appear to improve the performance significantly.

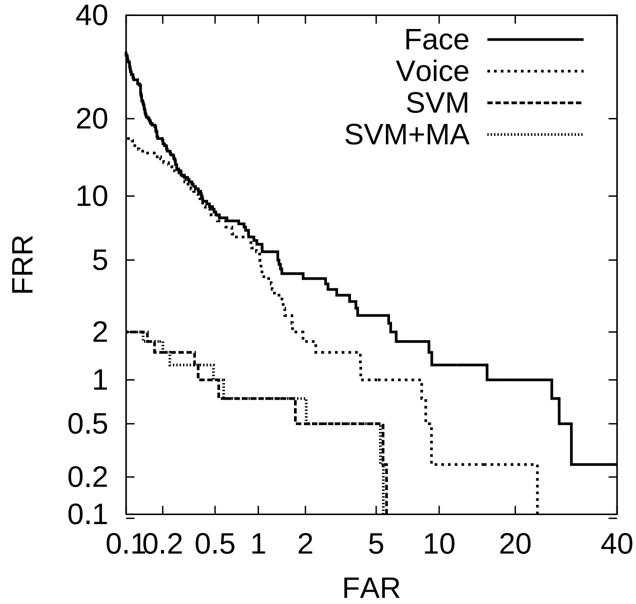

Fig. 3. Comparison of DET curves on the test set of XM2VTS configuration I. MLP+MA represents the DET curve obtained with a fusion algorithm using the model adequacy confidence measure.

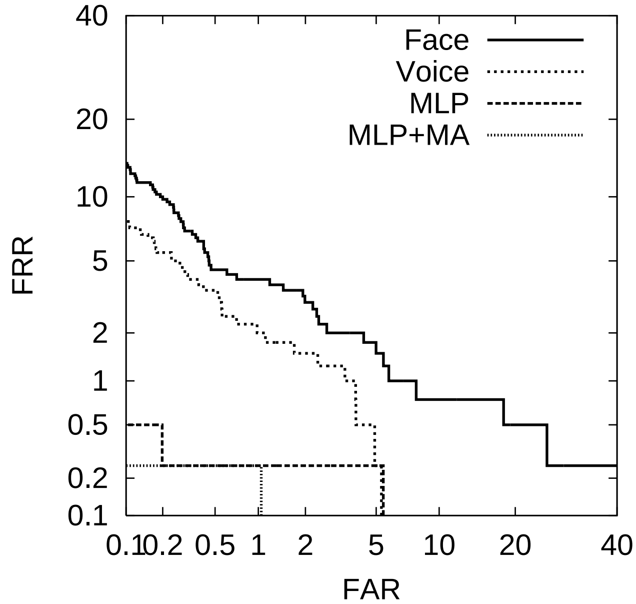

Fig. 4. Comparison of DET curves on the test set of XM2VTS configuration II. MLP+MA represents the DET curve obtained with a fusion algorithm using the model adequacy confidence measure.

In order to graphically see the improvement between the two unimodal verification systems, the best fusion algorithm (MLP) and the best confidence measure method (MLP using the Model Adequacy measure), Figures 3 and 4 show the corresponding DET curves for the test set of configurations I and II. Note that as all the fusion results presented in the previous tables were averaged over 10 experiments, the DET curves presented in these figures represent only one of these experiments for each method (the one which corresponded the most to the average).

5.4 Measuring Confidence of the Decisions

Another way to use confidence information is to compute a confidence measure of the fusion algorithm itself and verify *a posteriori* if this confidence measure was *reliable*. For this, we need a confidence measure that is both intuitive for the human decider (such as a probability to take the right decision) and easy to verify. The non parametric estimation method (section 4.2) seemed the most appropriate as it gives for each score a probability that the associated decision is false.

Hence, we computed the probabilities given by equation (13) for the test fusion decisions (using the baseline fusion algorithms, without any confidence measure used as input) and then computed the average absolute distance between the hypothesized probability and the real probability, as computed on the test set using the same idea of subspaces. Results are given in Table 4.

Fusion Method	Average Probability Error	
	Configuration I	Configuration II
SVMs	0.0021	0.0005
MLPs	0.0023	0.0009
Bayes	0.0021	0.0010

Table 4
Average distance between the predicted and obtained confidence estimation on the test set

These results show that the confidence measures produced by the non parametric method were on average different from the *real* confidence (as estimated *a posteriori* on the test set) by less than 0.25%, which lets us think that they are indeed reliable. Hence, while this method did not always produce useful information to enhance the performance of the fusion algorithm, it can still be used to produce precise confidence measures on the fusion decisions.

An interesting application of this measure consists in setting aside all scores having an unreliable confidence measure and perform authentication only on the other ones (the unreliable scores could then be referred to a human decider). Figure 5 shows how the HTER on configuration I could be enhanced by using various values of such a threshold, as well as the percentage of accesses that were set aside. This means that, for instance, we could enhance the HTER from 0.70 to less than 0.45 (a significant improvement) by setting aside 0.64% of the accesses (which represent in fact 719 accesses).

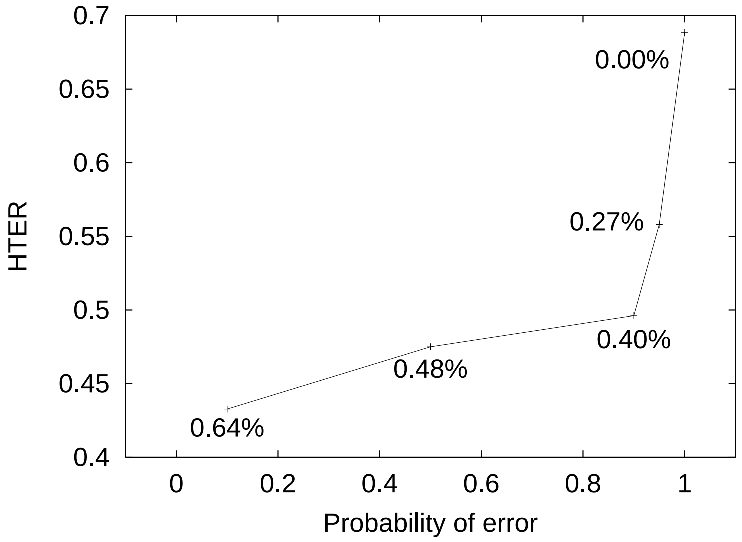

Fig. 5. HTER as a function of the probability of error, as given by the non parametric confidence measure in equation (13), and used to set aside a portion of the accesses. Results are on the test set of XM2VTS database (configuration I, test set). The numbers in the graph represent the percentage of accesses that were set aside.

6 Conclusion

In this paper we have presented three different approaches to estimate a confidence value over identity verification decisions. These approaches could either be used as additional inputs to a fusion algorithm or directly to measure the confidence of the final decisions. Experimental comparisons of fusion algorithms as well as confidence measures have been carried out using the XM2VTS benchmark database. Results showed that the Model Adequacy method was able to enhance the performance of the fusion algorithm systematically. On the other hand, a simple non parametric estimation of the confidence provided useful and reliable estimate of the confidence over the fusion decisions.

References

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

[2] R. Chellappa, C. Wilson, C. Barnes, Human and machine recognition of faces: A survey, Tech. Rep. CAR-TR-731, University of Maryland (1994).

[3] J. Zhang, Y. Yan, M. Lades, Face recognition: Eigenfaces, elastic matching, and neural nets, in: Proceedings of IEEE, Vol. 85, 1997, pp. 1422–1435.

[4] S. Marcel, S. Bengio, Improving face verification using skin color information, in: International Conference on Pattern Recognition (ICPR'2002), 2002.

[5] 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, E. Mayoraz, Comparison of face verification results on the XM2VTS database, in: A. Sanfeliu, J. J. Villanueva, M. Vanrell, R. Alqueraz, J. Crowley, Y. Shirai (Eds.), Proceedings of the 15th ICPR, Vol. 4, IEEE Computer Society Press, 2000, pp. 858–863.

[6] J. Kittler, R. Ghaderi, T. Windeatt, G. Matas, Face verification via ECOC, in: British Machine Vision Conference (BMVC01), 2001, pp. 593–602.

[7] 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), 1999, pp. 25–30.

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

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

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

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

[12] S. Ben-Yacoub, J. Lüettin, K. Jonsson, J. Matas, J. Kittler, Audio-visual person verification, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99), 1999, pp. 580–585.

[13] E. S. Bigün, J. Bigün, B. Duc, S. Fischer, Expert conciliation for multi modal person authentication systems by bayesian statistics, in: J. Bigün, G. Chollet, G. Borgefors (Eds.), Audio- and Video-based Biometric Person Authentication (AVBPA'97), 1997, pp. 291–300.

[14] S. Prabhakar, A. K. Jain, Decision-level fusion in biometric verification, in: Proceedings of the 2nd International Workshop on Multiple Classifier Systems (MCS), 2001, pp. 88–98.

[15] A. Ross, A. K. Jain, J. Z. Qian, Information fusion in biometrics, in: Proceedings of the 3rd International Conference on Audio- and Video-Based Person Authentication (AVBPA), 2001, pp. 354–359.

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

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

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

[19] A. Martin, M. Przybocki, The NIST 1999 speaker recognition evaluation - an overview, Digital Signal Processing 10 (2000) 1–18.

[20] J.-B. Pierrot, J. Lindberg, J. Koolwaaij, H.-P. Hutter, D. Genoud, M. Blomberg, 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, Vol. I, 1998, pp. 125–128.

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

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

[23] J. L. Gauvain, C.-H. Lee, Maximum a posteriori estimation for multivariate gaussian mixture observation of markov chains, in: IEEE Transactions on Speech Audio Processing, Vol. 2, 1994, pp. 291–298.

[24] L. Rabiner, B.-H. Juang, Fundamentals of speech recognition, 1st Edition, Prentice All, 1993.

[25] S. Bengio, J. Mariéthoz, S. Marcel, Evaluation of biometric technology on XM2VTS, Technical Report IDIAP-RR 01-21, IDIAP (2001).

[26] A. Kolmogorov, Sulla determinazione empirica di una leggi di distribuzione, G. Inst. Ital. Attuari 4, translated in English in *Breakthroughs in Statistics*, by Kotz and Johnson (editors), Springer-Verlag, 1992.

[27] T. Jaakkola, D. Haussler, Exploiting generative models in discriminative classifiers, in: M. Kearns, S. Solla, D. Cohn (Eds.), Advances in Neural Information Processing Systems 11: Proceedings of the 1998 Conference", MIT Press, 1999, pp. 487–493.

[28] J. Lüttin, G. Maître, Evaluation protocol for the extended M2VTS database (XM2VTSDB), Tech. Rep. RR-21, IDIAP (1998).

[29] J. Lüttin, Evaluation protocol for the the XM2FDB database (lausanne protocol), Tech. Rep. COM-05, IDIAP (1998).

[30] J. Hennebert, H. Melin, D. Petrovska, D. Genoud, POLYCOST: a telephone-speech database for speaker recognition, Speech Communication 31 (2000) 265–270.