



HAL
open science

Fast computation of the performance evaluation of biometric systems: application to multibiometric

Romain Giot, Mohamad El-Abed, Christophe Rosenberger

► To cite this version:

Romain Giot, Mohamad El-Abed, Christophe Rosenberger. Fast computation of the performance evaluation of biometric systems: application to multibiometric. *Future Generation Computer Systems*, 2013, pp.10.1016/j.future.2012.02.003. 10.1016/j.future.2012.02.003 . hal-00674526

HAL Id: hal-00674526

<https://hal.science/hal-00674526v1>

Submitted on 27 Feb 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Fast Computation of the Performance Evaluation of Biometric Systems: Application to Multibiometrics

Romain Giot*, Mohamad El-Abed*, Christophe Rosenberger*

GREYC Laboratory
ENSICAEN - University of Caen Basse Normandie - CNRS
6 Boulevard Maréchal Juin, 14000 Caen Cedex - France

Abstract

The performance evaluation of biometric systems is a crucial step when designing and evaluating such systems. The evaluation process uses the Equal Error Rate (EER) metric proposed by the International Organization for Standardization (ISO/IEC). The EER metric is a powerful metric which allows easily comparing and evaluating biometric systems. However, the computation time of the EER is, most of the time, very intensive. In this paper, we propose a fast method which computes an approximated value of the EER. We illustrate the benefit of the proposed method on two applications: the computing of non parametric confidence intervals and the use of genetic algorithms to compute the parameters of fusion functions. Experimental results show the superiority of the proposed EER approximation method in term of computing time, and the interest of its use to reduce the learning of parameters with genetic algorithms. The proposed method opens new perspectives for the development of secure multibiometrics systems with speeding up their computation time.

Keywords:

Biometrics, Authentication, Error Estimation, Access Control

1. Introduction

Biometrics [1] is a technology allowing to recognize people through various personal factors. It is an active research field which design new biometric traits from time to time (like finger knuckle recognition [2]). We can classify the various biometric modalities among three main families:

*Corresponding author

Email addresses: `romain.giot@ensicaen.fr` (Romain Giot),
`mohamad.elabed@ensicaen.fr` (Mohamad El-Abed), `christophe.rosenberger@ensicaen.fr`
(Christophe Rosenberger)

- *Biological*: the recognition is based on the analysis of biological data linked to an individual (*e.g.*, DNA, EEG analysis, ...).
- *Behavioural*: the recognition is based on the analysis of the behaviour of an individual while he is performing a specific task (*e.g.*, signature dynamics, gait, ...).
- *Morphological*: the recognition is based on the recognition of different physical patterns, which are, in general, permanent and unique (*e.g.*, fingerprint, face recognition, ...).

It is mandatory to evaluate these biometric systems in order to quantify their performance and compare them.

These biometric systems must be evaluated in order to compare them, or to quantify their performance. To evaluate a biometric system, a database must be acquired (or a common public dataset must be used). This database must contain as many users as possible to provide a large number of captures of their biometric data. These data are separated into two different sets:

- *the learning* set which serves to compute the biometric reference of each user
- *the validating* set which serves to compute their performance.

When comparing test samples to biometric references, we obtain two different kinds of scores:

- *the intrascores* represent comparison scores between the biometric reference (computed thanks to the learning set) of an individual and biometric query samples (contained in the validating set)
- *the interscores* represent comparison scores between the biometric reference of an individual and the biometric query samples of the other individuals.

From these two sets of scores, we can compute various error rates, from which the EER is one functioning point which represents a very interesting error rate often used to compare biometric systems. In order to have reliable results, it is necessary to evaluate the performance of biometric system with huge datasets. These huge datasets produce numbers of scores. As the time to evaluate the performance of a biometric system depends on the quantity of available scores, we can see that evaluation may become very long on these large datasets. In this paper, we present a very fast way to compute this error rate, as well as its confidence interval in a non parametric way, on different datasets of the literature.

Nevertheless, there will always be users for which one modality (or method applied to this modality) will give bad results. These low performances can be implied by different facts: the quality of the capture, the acquisition conditions, or the individual itself. Biometric multi-modality (or multibiometrics) allows to

compensate this problem while obtaining better biometric performances (*i.e.*, better security by accepting less impostors, and better usability by rejecting less genuine users) by expecting that the errors of the different modalities are not correlated. So, the aim of multibiometrics is to protect logical or physical access to a resource by using different biometric captures. We can find different types of multibiometrics systems. Most of them are listed in [3], they use:

1. Different sensors of the same modality (*i.e.*, capacitive or resistive sensors for fingerprint acquisition);
2. Different representations of the same capture (*i.e.*, use of points of interest or texture);
3. Different biometric modalities (*i.e.*, face and fingerprint);
4. Several instances of the same modality (*i.e.*, left and right eye for iris recognition);
5. Multiple captures (*i.e.*, 25 images per second in a video used for face recognition);
6. An hybrid system composed of the association of the previous ones.

In the proposed study, we are interested in the first four kinds of multimodality. We also present in this paper, a new multibiometrics approach using various fusion functions parametrized by genetic algorithms using a fast EER (Equal Error Rate) computing method to speed up the fitness evaluation.

This paper is related to high performance computing, because algorithms are designed to work in an infrastructure managing the biometric authentication of millions of individuals (*i.e.*, border access control, logical access control to webservices). To improve the recognition rate of biometric systems, it is necessary to regularly update the biometric reference to take into account intra class variability. With the proposed approach, the time taken to update the biometric reference would be lowered. The faster is the proposed method, the more we can launch the updating process (or the more users we can add to the process). We also propose an adaptation of the proposed EER computing method which gives confidence intervals in a non parametric way (*i.e.*, by computing the EER several times through a bootstrapping method). The confidence intervals are computed on a single CPU, on several CPUs on the same machine and on several machines.

The main hints of the papers are:

- the proposition of a new method to approximate the EER and its confidence interval in a fast way
- the proposition of two original functions for multibiometrics fusion.

The plan is organized as following. Section 2 presents the background of the proposed work. Section 3 presents the proposed method for computing the approximated value of the EER and its confidence interval. Section 4 validates them. Section 5 presents the proposed multibiometrics fusion functions and their performance in term of biometric recognition and computation time against the baseline. Section 6 gives perspectives and conclusions of this paper.

2. Background

2.1. Evaluation of Biometric Systems

Despite the obvious advantages of this technology in enhancing and facilitating the authentication process, its proliferation is still not as much as attended [4]. As argued in the previous section, biometric systems present several drawbacks in terms of precision, acceptability, quality and security. Hence, evaluating biometric systems is considered as a challenge in this research field. Nowadays, several works have been done in the literature to evaluate such systems. Evaluating biometric systems is generally realized within three aspects as illustrated in figure 1: usability, data quality and security.

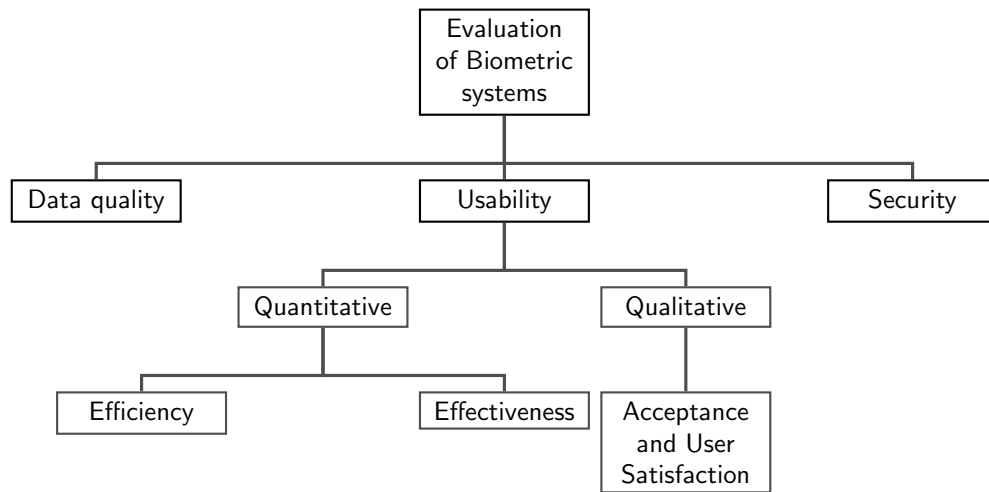


Figure 1: Evaluation aspects of Biometric Systems.

2.1.1. Usability

According to the International Organization for Standardization ISO 13407:1999 [5], usability is defined as “*The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use*”.

- *Efficiency* which means that users must be able to accomplish the tasks easily and in a timely manner. It is generally measured as task time;
- *Effectiveness* which means that users are able to complete the desired tasks without too much effort. It is generally measured by common metrics include completion rate and number of errors such failure-to-enroll rate (FTE) [6];
- *User satisfaction* which measures users’ acceptance and satisfaction regarding the system. It is generally measured by studying several properties such as easiness to use, trust in the system, *etc.* The acceptability

of biometric systems is affected by several factors. According to [7], some members of the human-computer interaction (HCI) community believe that interfaces of security systems do not reflect good thinking in terms of creating a system that is easy to use, while maintaining an acceptable level of security. Existing works [8, 9] show also that there is a potential concern about the misuse of personal data (*i.e.*, templates) which is seen as violating users' privacy and civil liberties. Moreover, one of our previous work [10] shows the necessity of taking into account users' acceptance and satisfaction when designing and evaluating biometric systems. More generally speaking, even if the performance of a biometric system outperformed another one, this will not necessarily mean that it will be more operational or acceptable;

2.1.2. Data quality

It measures the quality of the biometric raw data [11, 12]. Low quality samples increase the enrollment failure rate, and decrease system performance. Therefore, quality assessment is considered as a crucial factor required in both the enrollment and verification phases. Using quality information, the bad quality samples can be removed during enrollment or rejected during verification. Such information could also be used in soft biometrics or multimodal approaches [13]. Such type of assessment is generally used to quantify biometric sensors, and could be also used to enhance system performance;

2.1.3. Security

It measures the robustness of a biometric system (algorithms, architectures and devices) against attacks. Many works in the literature [14, 15, 16] show the vulnerabilities of biometric systems which can considerably decrease their security. Hence, the evaluation of biometric systems in terms of security is considered as an important factor to ensure its functionality. The International Organization for Standardization ISO/IEC FCD 19792 [17] addresses the aspects of security evaluation of such systems. The report presents an overview of biometric systems vulnerabilities and provide some recommendations to be taking into account during the evaluation process. Nowadays, only few partial security analysis studies with relation to biometric authentication systems exist. According to ISO/IEC FCD 19792 [17], the security evaluation of biometric systems is generally divided into two complementary assessments: 1) assessment of the biometric system (devices and algorithms) and 2) assessment of the environmental (for example, is the system is used indoor or outdoor?) and operational conditions (for example, tasks done by system administrators to ensure that the claimed identities during enrolment of the users are valid). A type-1 security assessment method is presented in a personal previous work [18]. The proposed method has shown its efficiency in evaluating and comparing biometric systems.

2.2. Performance Evaluation of Biometric Systems

The performance evaluation of biometric systems is now carefully considered in biometric research area. We need a reliable evaluation methodology in order to put into obviousness the benefit of a new biometric system. Nowadays, many efforts have been done to achieve this objective. We present in section 2.2.1 an overview of the performance metrics, followed by the research benchmarks in biometrics as an illustration of the evaluation methodologies used in the literature for the comparison of biometric systems.

2.2.1. Performance metrics

By contrast to traditional methods, biometric systems do not provide a cent per cent reliable answer, and it is quite impossible to obtain such a response. The comparison result between the acquired biometric sample and its corresponding stored template is illustrated by a distance score. If the score is lower than the predefined decision threshold, then the system accepts the claimant, otherwise he is rejected. This threshold is defined according to the security level required by the application. Figure 3 illustrates the theoretical distribution of the genuine and impostor scores. This figure shows that errors depend from the used threshold. Hence, it is important to quantify the performance of biometric systems. The International Organization for Standardization ISO/IEC 19795-1 [6] proposes several statistical metrics to characterize the performance of a biometric system such as:

- *Failure-to-enroll rate (FTE)*: proportion of the user population for whom the biometric system fails to capture or extract usable information from biometric sample;
- *Failure-to-acquire rate (FTA)*: proportion of verification or identification attempts for which a biometric system is unable to capture a sample or locate an image or signal of sufficient quality;
- *False Acceptation Rate (FAR)* and *False Rejection Rate (FRR)*: FAR is the proportion of impostors that are accepted by the biometric system, while the FRR is the proportion of authentic users that are incorrectly denied. The computation of these error rates is based on the comparison of the scores against a threshold (the direction of the comparison is reversed if the scores represent similarities instead of distances). FRR and FAR are respectively computed (in the case of a distance score) as in (1) and (2), where $intra_i$ (respectively $inter_i$) means the intra score at position i in the set of intra score (respectively inter score at position i) and $Card(set)$ is the cardinal of the set in argument, thr is the decision threshold, and $\mathbf{1}$ is the indicator function.

$$FRR = \frac{\sum_{score \in intra} \mathbf{1}\{score > thr\}}{Card(intra)} \quad (1)$$

$$FAR = \frac{\sum_{score \in intra} \mathbb{1}\{score \leq thr\}}{Card(intra)} \quad (2)$$

- *Receiver operating characteristic (ROC) curve*: the ROC curve is obtained by computing the couple of (FAR, FRR) for each tested threshold. It plots the FRR versus the FAR. The aim of this curve is to present the tradeoff between FAR and FRR and to have a quick overview of the system performance and security for all the parameters configurations.
- *Equal error rate (EER)*: it is the value where both errors rates, FAR and FRR, are equals (*i.e.*, FAR = FRR). It constitutes a good indicator, and the most used, to evaluate and compare biometric systems. In other words, lower the EER value is, higher the accuracy of the system. Using the ROC curve, the EER is computed by selecting the couple of (FAR, FRR) having the smallest absolute difference (3) at the given threshold τ :

$$\tau = \underset{\tau}{\operatorname{argmin}}(abs(FAR_{\tau} - FRR_{\tau}), \forall_{\tau \in Card\{ROC\}})$$

and returning their average (3):

$$EER = \frac{FAR_{\tau} + FRR_{\tau}}{2} \quad (3)$$

By this way, we have obtained the best approaching EER value with the smallest precision error. The classical EER computing algorithm is presented in the Figure 2¹. From Figure 2, we can see that the complexity is in $O(n * m)$ with n the number of thresholds held in the computation, and, m the number of scores in the dataset. As it is impossible to reduce m , we have to find a better method which reduces n . We did not find, in the literature, methods allowing to reduce computation time in order to obtain this EER.

2.2.2. Biometrics Datasets

A public dataset allows researchers to test their algorithm and compare them with those from the state of the art. It takes a lot of time and energy to build a large and significant dataset. It is very convenient to download one for research purposes. We present in this section an overview of the used datasets in this paper. Table 1 presents a summary of these datasets.

¹another, slower, way of computing would be to test each unique score of the intrascores and interscores sets, but this would held a too important number of iterations. We named it “whole” later in the paper.


```

ROC ← []
EER ← 1.0
DIFF ← 1.0
START ← min(scores)
END ← max(scores)
for τ START to END in N steps do
  FAR ← compute FAR for τ
  FRR ← compute FRR for τ
  append (FAR, FRR) to ROC
  if abs(FAR - FRR) < DIFF then
    DIFF ← abs(FAR - FRR)
    EER ← (FAR + FRR)/2
  end if
end for
return EER, ROC

```

Figure 2: Classical EER computing algorithm.

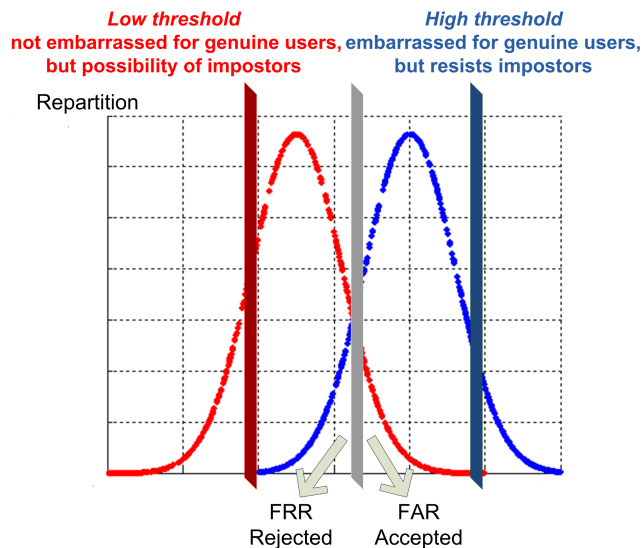


Figure 3: Distribution of genuine users and impostor scores.

- **Biometric Scores Set - Release 1 (BSSR1)**

The BSSR1 [19] database is an ensemble of scores sets from different biometric systems. In this study, we are interested in the subset containing the scores of two facial recognition systems and the two scores of a fingerprint recognition system applied to two different fingers for 512 users. This database has been used many times in the literature [20, 21].

- **BANCA**

The second database is a subset of scores produced from the BANCA database [22]. The selected scores correspond to the following one labelled:

1. IDIAP_voice_gmm_auto_scale_25_100_pca.scores

2. SURREY_face_nc_man_scale_100.scores
3. SURREY_face_svm_man_scale.0.13.scores
4. UC3M_voice_gmm_auto_scale_10_100.scores

The database as two subsets G1 and G2. G1 set is used as the learning set, while G2 set is used as the validation set.

- **PRIVATE**

The last database is a chimeric one we have created for this purpose by combining two public biometric template databases: the AR [23] for the facial recognition and the GREYC keystroke [24] for keystroke dynamics [25, 26]. The AR database is composed of frontal facial images of 126 individuals under different facial expressions, illumination conditions or occlusions. These images have been taken during two different sessions with 13 captures per session. The GREYC keystroke contains the captures on several sessions on two months of 133 individuals. Users were asked to type the password "greyc laboratory" 6 times on a laptop and 6 times on an USB keyboard by interlacing the typings (one time on a keyboard, one time on another). We have selected the first 100 individual of the AR database and we have associated each of these individuals to another one in a subset of the GREYC keystroke database having 5 sessions of captures. We then used the 10 first captures to create the biometric reference of each user and the 16 others to compute the intra and inter scores. These scores have been computed by using two different methods for the face recognition and two other ones for the keystroke dynamics.

Table 1: Summary of the different databases used to validate the proposed method

Nb of	BSSR1	PRIVATE	BANCA
users	512	100	208
intra tuples	512	1600	467
inter tuples	261632	158400	624
items/tuples	4	5	4

2.3. Multibiometrics

We focus in this part on the state of the art on multimodal systems involving biometric modalities usable for all computers (keystroke, face, voice...). The scores fusion is the main process in multimodal systems. It can be operated on the scores provided by algorithms or in the templates themselves [27]. In the first case, it is necessary to normalize the different scores as they may not evolve in the same range. Different methods can be used for doing this, and the most efficient methods are *zscore*, *tanh* and *minmax* [28]. Different kinds of fusion methods have been applied on biometric systems. The fusion can be

done with multiple algorithms of the same modality. For example, in [29], three different keystroke dynamics implementations are fused with an improvement of the EER, but less than 40 users are involved in the database. In [30], two keystroke dynamics systems are fused together by using weighted sums for 50 users, but no information on the weight computing is provided. The fusion can also be done within different modalities in order to improve the authentication process. In [31], authors use both face and fingerprint recognition, the impact of error rate reduction is used to reduce the error when adapting the user’s biometric reference. There is only one paper (to our knowledge) on keystroke dynamics fusion with another kind of biometric modality (voice recognition): it is presented in [32], but only 10 users are involved in the experiment. In [33], multi-modality is done on fingerprints, speech, and face images on 50 individuals. Fusion has been done with SVM [34] with good improvements, especially, when using user specific classifiers.

Very few multimodal systems have been proposed for being used in classical computers and the published ones have been validated on small databases. In order to contribute to solve this problem, we propose a new approach in the following section.

3. Fast EER and Confidence Intervals Estimation

We propose a kind of dichotomic EER computing function, in order to quickly approximate its value. Thanks to this computing speed up, we can use it in time consuming applications. Finally, we present a confidence interval computing method based on our approximated EER calculation associated to parallel and distributed computing.

3.1. EER Estimation

Computation time to get the EER can be quite important. When the EER value needs to be computed a lot of time, it is necessary to use a faster way than the standard one. In the biometric community, the shape of the ROC curve always follows the same pattern: it is a monotonically decreasing function (when we present FRR against FAR, or increasing when we present 1-FRR against FAR) and the EER value is the curve’s point having $x_{ROC} = y_{ROC}$ (or $FAR = FRR$). Thanks to this fact, the curve symbolising the difference of y_{ROC} against x_{ROC} is also a monotonically decreasing function from 1 to -1 , where the point at $y_{DIFF} = 0$ represents the EER (and its value is x_{DIFF} because $x_{ROC} = y_{ROC}$ or $FAR = FRR$). With these information, we know that to get the EERs, we need to find the x_{DIFF} for which y_{DIFF} is the closest as possible to zero. An analogy with the classical EER computing, would be to incrementally compute y_{DIFF} for each threshold by increasing order and stop when y_{DIFF} changes of sign. By this way, we can expect to do half thresholds comparisons than with the classical way if scores are correctly distributed. A clever way is to use something equivalent to a divide and conquer algorithm like the binary search and obtain a mean complexity closer to $O(\log(n))$. That is why we have implemented a polytomous version of EER computing:

1. We chose i thresholds linearly distributed on the scores sets
2. For each threshold t among the i thresholds, we compute the FAR and FRR values (FAR_t, FRR_t)
3. We take the two following thresholds $t1$ and $t2$ having $sign(FRR_{t1} - FAR_{t1})$ different of $sign(FRR_{t2} - FAR_{t2})$
4. We repeat step 2 with selecting i thresholds between $t1$ and $t2$ included while $FRR_{t1} - FAR_{t1}$ does not reach the attended precision.

By this way, the number of threshold comparisons is far smaller than in the classical way. Its complexity analysis is not an easy task because it depends both on the attended precision and the choice of i . It can be estimated as $O(\log(N))$.

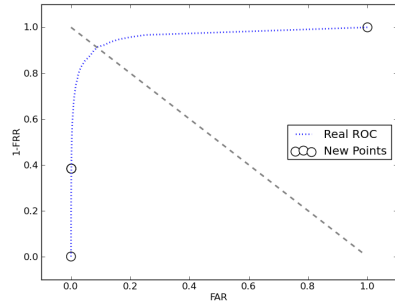
Figure 4 presents the algorithm while Figure 5 illustrates it by showing the different iterations until getting the EER value with a real world dataset. We have chosen $i = 5$ points to compute during each iteration. The EER is obtained in five iterations. Circle symbols present the points computed at the actual iteration, triangle symbols present the points computed at the previous iterations, and the dotted curve presents the ROC curve if all the points are computed. Very few points are computed to obtained the EER value. Figure 5f presents the real ROC curve and the ROC curve obtained with the proposed method. We can see that even if we obtain an approximated version of the real ROC curve, it is really similar around the EER value (cross with the dotted lined).

```

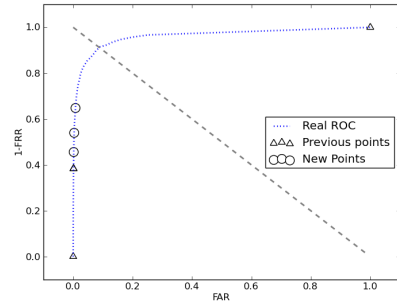
ROC ← []
CACHE ← {}
START ← min(scores)
END ← max(scores)
while True do
  for THRESHOLD from START to END in N steps do
    SDIFF ← []
    THRESHOLDS ← []
    if not empty CACHE[THRESHOLD] then
      FAR, FRR ← CACHE[THRESHOLD]
    else
      FAR ← compute FAR for THRESHOLD
      FRR ← compute FRR for THRESHOLD
      append (FAR, FRR) to ROC
      CACHE[THRESHOLD] ← (FAR, FRR)
    end if
    if abs(FAR - FRR) < PRECISION then
      EER ← (FAR + FRR)/2
      return EER, ROC
    end if
    append FAR - FRR to SDIFF
    append THRESHOLD to THRESHOLDS
  end for
  PSTART ← -1
  PEND ← -1
  for PIVOT = 0 to STEPS - 1 do
    if sign(SDIFF[PIVOT]) ≠ sign(SDIFF[PIVOT + 1]) then
      PSTART ← PIVOT
      PEND ← PIVOT + 1
      break
    end if
  end for {PSTART and PEND are set}
  START ← THRESHOLDS[PSTART]
  END ← THRESHOLDS[PEND]
end while

```

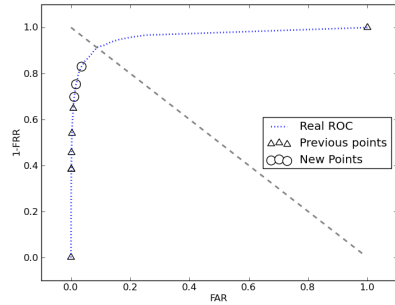
Figure 4: Fast EER Computing Algorithm



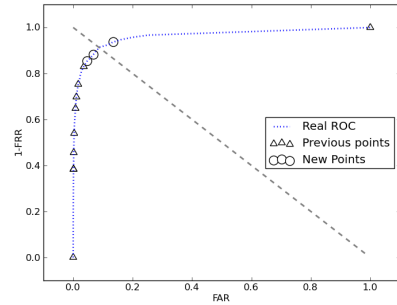
(a) Iteration 1



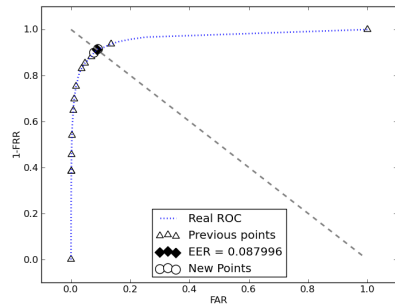
(b) Iteration 2



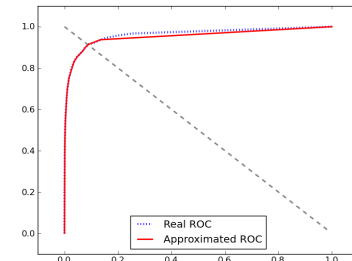
(c) Iteration 3



(d) Iteration 4



(e) Iteration 5



(f) ROC curve

Figure 5: Points computed by the proposed algorithm when $i = 5$. In this case, the EER value is found in five iterations. Each image represents an iteration, with: the real ROC curve, the points computed at the iteration and the points computed at the previous iteration (different thresholds may produce the same points).

3.2. Confidence Intervals Estimation

We also provide a method to compute the confidence interval of the EER value. It is based on a bootstrapping method and can be used in a parallelized way.

3.2.1. Bootstrapping

It is interesting to give a confidence interval of an EER value, because we are not totally sure of its value. One way is to obtain this confidence interval parametrically, but it requires to have strong hypothesis on the function of the EER value (the scores come from independent and identically distributed variables, even for the scores of the same user). As such assumption is too strict (and probably false), it is possible to use non parametric confidence intervals. One of these non parametric methods is called “bootstrap” [35]. Such method is often used when the distribution is unknown or the number of samples is too low to correctly estimates the interval. The main aim is to re-sample the scores several times, and, compute the EER value for each of these re-sampling. The bootstrapping method works as following:

1. Use the *intra* and *inter* scores to compute the EER $\hat{\chi}$.
2. Resample K times the *intra* and *inter* scores and store them in $intra^i$ and $inter^i$ ($0 < i \leq K$).
 - Generate the resampled $intra^i$ scores by sampling $Card(intra)$ scores with replacement from *intra*.
 - Generate the $inter^i$ scores by sampling $Card(inter)$ scores with replacement from *inter*.
3. Compute the K EERs (χ^i) for each couple of $intra^i$ and $inter^i$ ($0 < i \leq K$).
4. Store the K residuals $e^i = \hat{\chi} - \chi^i$.
5. The $100(1 - \alpha)\%$ confidence interval of the EER is formed by taking the interval from $\hat{\chi} - e_{Upper}$ to $\hat{\chi} - e_{Lower}$, with e_{Lower} and e_{Upper} which respectively represent the $\alpha/2$ th and the $1 - \alpha/2$ th percentiles of the distribution of e .

Figure 6 presents the residuals of one run of the bootstrapped method on a real world dataset with the lower and upper limits used to compute the confidence interval. We can see that the consuming part of this algorithm is the fact to compute K times the EER. As all the EER computing is totally independent from each other, the K computations can be done in a parallel way.

3.2.2. Parallelization

We propose three different ways to compute the confidence interval:

- *Single*. The single version consists in doing all the computations in a sequential manner (*cf.* figure 7a). The EER with the original scores is computed. The whole set of resampled scores is created. The EER of each new distribution is computed. The confidence interval is computed from the results.

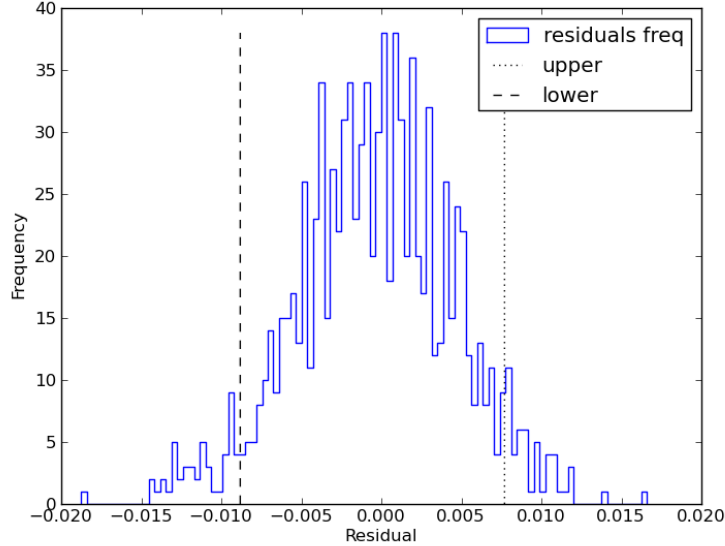


Figure 6: Histogram of residuals and $\alpha/2$ and $1-\alpha/2$ percentiles for a confidence interval at 90%.

- *Parallel.* The parallel version consists in using the several cores or processors on the computer used for the computation (*cf.* figure 7b). In this case, several EERs may be computed at the same time (in the better case, with a computer having n processing units, we can compute n different results at the same time). The procedure is the following: the EER with the original scores is computed. The whole set of resampled scores is created. The main program distributes the K EERs computations on the different processing units. Each processing unit computes an EER and returns the results, until the main program stops to send it new data. The results are merged together. The confidence interval is computed from the results.
- *Distributed.* The distributed version consists in using several computers to improve the computation (*cf.* figure 7c). The computation is done in a parallelized way on each computer. In this case, much more EERs can be computed at the same time. The main program generates a set $S = \{S_1, \dots, S_T\}$ of T values symbolising T subworks, where the subwork T_i must compute S_i EERs (thus $\sum S = K$). The procedure is the following: the EER with the original scores is computed. The main program sends the intra and inter scores on each worker (*i.e.*, a computer). The main program distributes the T numbers to each worker. Each time a work receive such number (S_i), it computes the S_i resamples sets. Then, it computes (using the Parallelized way) the S_i EERs by distributing them

on its processing units. It merges the S_i EERs together and send them to the main program. The main program merges all the results together. The confidence interval is computed from the results.

We can see that the three schemes are totally different. The parallelized version may be used on all recent computers which have several processing units, while the distributed version needs to use several computers connected through a network.

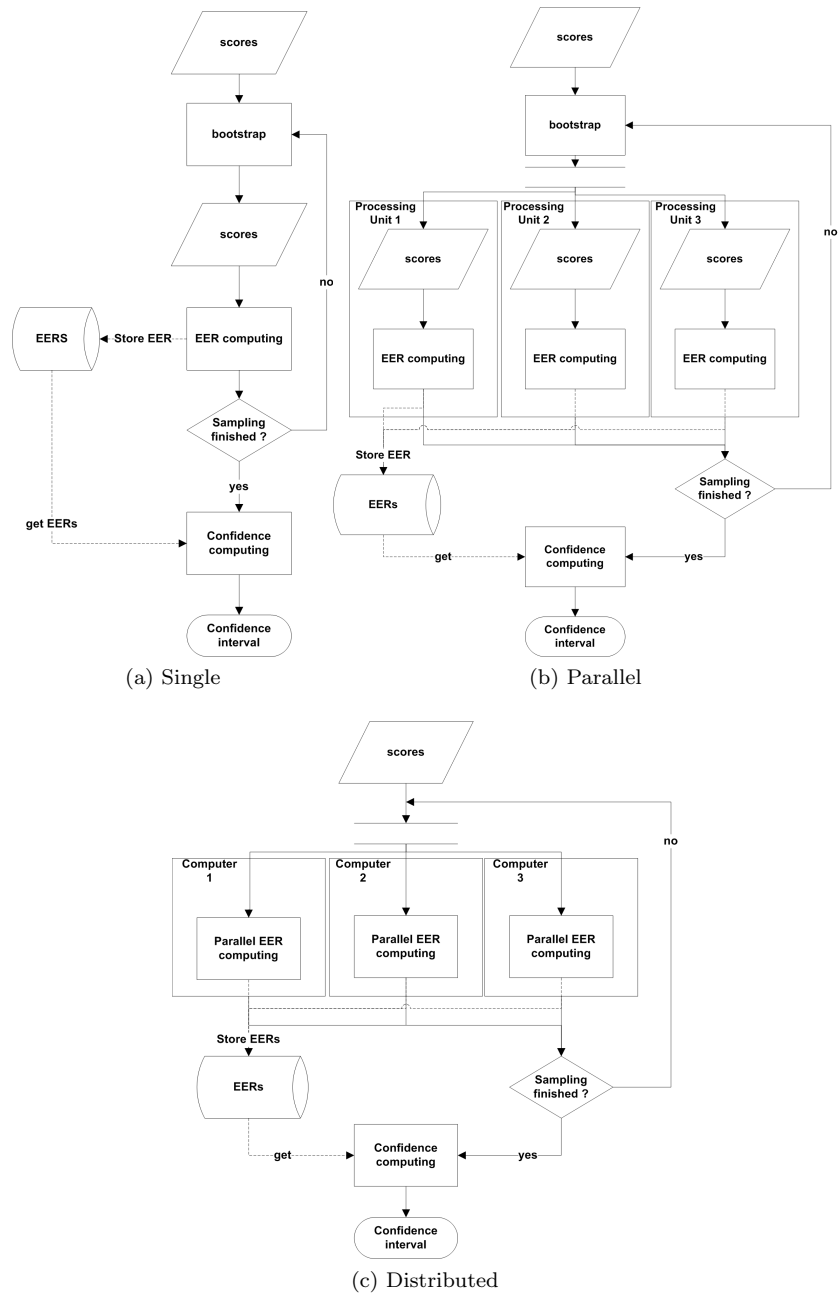


Figure 7: Different architectures to compute the confidence interval.

4. Protocol

4.1. Databases Sets

In order to do these evaluations, we have used three different biometric databases presented in section 2.2.2.

4.2. Evaluation of the EER Computing

The two different algorithms for EER computing have been run on five different sets of scores (three of keystroke dynamics and two of face recognition, generated with the PRIVATE database) with various parameters. We call *classic* the classical way of computing the EER and *polyto* the proposed version of the algorithm. The classic way is tested by using 50, 100, 500 and 1000 steps to compute the EER. The polytomous way is tested by using between 3 and 7 steps and a precision of 0.01, 0.005 and 0.003. The aim of these tests is to compare how the proposed method performs better than the classical one, and what are its best parameters.

4.3. Utility for Non-parametric Confidence Interval

Confidence intervals are also an interesting information on the performance of a system. The properties of the score distribution may forbid the use of parametric methods to compute it. This is where the bootstrap method helps us by computing several times the EER with resampling methods. We have tested the computation time of confidence interval computing for two systems of each database, which give us six different systems. We have computed the EER value under three different ways: the polytomous one, the classical one (with 1000 steps), and another we called whole. The whole method is similar to the classical one, except that it uses all the possible scores present in the intra inter scores arrays as thresholds, instead of artificially generating them in a predefined interval (thus, there may be far more thresholds than in the classic method, or far less, depending on the number of available scores in the database). The results are then validated with confidence intervals. The test scripts were written with the Python language. The EER computing methods and the resampling methods have been compiled in machine code thanks to the Cython [36] framework (which speed up computation time). The parallelization is done with the joblib [37] library. The distributed version is done by using the task management provided by Ipython [38]. The standard and parallelized versions have been launched on a recent Linux machine with an Intel® Core™ i5 with 4 cores at 3.20GHz and 4Gb of RAM. Four processes are launched at the same time. For the distributed machine, the orchestrator is the same machine. It distributes the jobs on three multicores machines (itself, another similar machine and an another machine with an Intel® Xeon® with 8 cores at 2.27GHz and 4Gb of RAM. The controller sends two more jobs last machine machine).

4.4. Experimental Results

4.4.1. Evaluation of the EER Computing

Table 2 presents the results obtained within the first tested biometric system. We present the name of the method, the error of precision while computing the EER, the computation time in milliseconds and the number of comparisons involved (each comparison corresponds to the comparison of a threshold against the whole set of intra and inter scores). The real computation time taken by a comparison is given in (4), where n is the number of thresholds to compare, A is the timing to do a comparison and B and C depends on the algorithm.

$$T = n * (A * (Card\{intra\} + Card\{inter\}) + B) + C \quad (4)$$

We can see that the computation time is highly related to the number of comparisons and the size of the score set. Using the Kruskal-Wallis test at a confidence degree equals to 95%, the proposed method significantly outperformed the classical method in terms of errors (with a p value = 0.0305) and computation time (with a p value = 0.002562). The obtained results are slightly similar for the five tested biometrics modalities. We can observe that, in the classic method, using 50 steps gives not enough precise results, while using 1000 gives a very good precision, but is really time consuming; depending on the dataset, 500 steps seems to be a good compromise between precision error and computation time. In all the polytomous configurations, the computation time is far better than the fastest classic method (50 steps) while having a greatest precision. This precision is always better than the classic method with 100 steps and approach or is better than the precision in 1000 steps. This gain of time is due to the lowest number of involved comparisons. In an n steps classical computing, we need to check n thresholds, while in the polytomous way this number depends both on the dataset and the required precision: with our dataset, it can vary from 8 to 35 which is always lower than 50. As the computation time depends only on this value, we can say that the fastest algorithms are the one having the smallest number of tests to complete.

Based on the number of comparisons (and the timing computation), Table 3 presents the best results for each modality (when several methods return the same number of iterations, the most precise is chosen).

We can argue that the proposed method is better, both in terms of speed and precision error, than the classical way of computing. Based on the results of our dataset, the configuration using 3 steps and a precision of 0.010 seems to be the best compromise between speed and precision. We can now argue that the proposed EER computation will speed up genetic algorithms using the EER as fitness function.

4.4.2. Utility for Non-parametric Confidence Interval

All the methods under all the implementations give similar confidence intervals. We do not discuss on this point, because we are only interested in computation time. Table 4 gives for each method, under each implementation

Table 2: Comparison of the Different EER Computing Methods And Configurations On The First Test Set. LABEL presents the used method. ERROR is the difference between FAR and FRR values. TIME is the time needed to compute the EER value. COMP. is the number of threshold comparison done.

LABEL	ERROR (%)	TIME (ms.)	COMP.
classic_50	8.37	459	50
classic_100	4.13	940	100
classic_500	0.20	4700	500
classic_1000	0.20	9310	1000
polyto_3_0.010	0.30	110	11
polyto_3_0.005	0.07	139	14
polyto_3_0.003	0.07	140	14
polyto_4_0.010	0.40	140	15
polyto_4_0.005	0.20	149	16
polyto_4_0.003	0.10	169	18
polyto_5_0.010	0.30	150	16
polyto_5_0.005	0.07	190	20
polyto_5_0.003	0.07	179	20
polyto_6_0.010	0.40	140	15
polyto_6_0.005	0.10	179	19
polyto_6_0.003	0.10	179	19
polyto_7_0.010	0.07	190	21
polyto_7_0.005	0.07	190	21
polyto_7_0.003	0.07	200	21

Table 3: Fastest EER Computing Parameters For Each Modality

DB	LABEL	ERROR (%)	TIME	COMP.
1	polyto_3_0.010	0.30	110	11
2	polyto_3_0.010	0.05	50	5
3	polyto_6_0.003	0.09	60	7
4	polyto_3_0.010	0.14	89	10
5	polyto_4_0.010	0.29	70	7

the mean value of the computation time for all the six different biometric systems. We can observe that, in average, the polytomous version seems far more faster than the other methods (classic and whole). The distributed implementation seems also more faster than the other implementations (Parallel, Single). Using the Kruskal-Wallis test at a confidence degree equals to 95%, the computation time of the proposed method is significantly faster than both classical (p value = 0.00651) and whole (p value = 0.004407) schemes. There was no significant difference of computation time between classical and whole schemes (p value = 0.8002).

Table 4: Summary of the computation time in seconds. Time are averaged on all the set of scores

	Polytomous	Classic	Whole	Mean
Distributed	14.93	94.25	1009.28	372.82
Parallel	14.95	276.60	3154.32	1148.63
Single	18.83	523.79	6733.68	2425.43
Mean	16.24	298.22	3632.43	

4.4.3. Discussion

We have demonstrated the superiority of our EER estimation method against the classical method concerning the computation time. However, the method stops when the required precision is obtained. As the method is iterative, it is not parallelizable when we want only a simple EER. However using a grid computing method greatly improves the computation of confidence intervals.

5. Application to Multibiometrics Fusion Function Configuration

We propose a biometric fusion system based on the generation of a fusion function parametrized by a genetic algorithm and a fast method to compute the EER (which is used as fitness function) in order to speed the computing time of the genetic algorithm.

5.1. Method

We have tested three different kinds of score fusion methods which parameters are automatically set by genetic algorithms [39]. These functions are presented in (5), (6) and (7) where n is the number of available scores (*i.e.*, the number of biometric systems involved in the fusion process), w_i the multiplication weight of score i , s_i the score i and x_i the weight of exponent of score i . (5) is the commonly used weighted sum (note that in this version, the sum of the weights is not equal to 1), while the two others, to our knowledge, have never been used in multibiometrics. We have empirically designed them in order to give more weights to higher scores.

$$ga1 = \sum_{i=0}^n w_i * s_i \quad (5)$$

$$ga2 = \prod_{i=0}^n s_i^{x_i} \quad (6)$$

$$ga3 = \sum_{i=0}^n w_i * s_i^{x_i} \quad (7)$$

The aim of the genetic algorithm is to optimize the parameters of each function in order to obtain the best fusion function. Each parameter (the w_i

Table 5: Configuration of the Genetic Algorithms

Parameter	Value
Population	5000
Generations	500
Chromosome signification	weights and powers of the fusion functions
Chromosome values interval	$[-10; 10]$
Fitness	polytomous EER on the generated function
Selection	normalized genetic selection (probability of 0.9)
Mutation	boundary, multi non uniform, non uniform, uniform
Cross-over	Heuristic Crossover
Elitism	True

and x_i) is stored in a chromosome of real numbers. The fitness function is the same for the three genetic algorithms. It is processed in two steps:

- *fusion*: The generated function (5), (6) or (7) are evaluated on the whole set of scores;
- *error computing*: The EER is computed on the result of the fusion. We use the polytomous version of computing in order to highly speed up the total computation time.

5.2. Experimental Protocol

5.2.1. Design of Fusion Functions

Table 5 presents the parameters of the genetic algorithms. The genetic algorithms have been trained on a learning set composed of half of the intrascores and half of the interscores of a database and they have been verified with a validation set composed of the others scores. The three databases have been used separately.

The generated functions are compared to three methods of the state of the art: *sum*, *mul* and *min*, they have been explored in [28, 40]. Table 6 presents, for each database, the EER value of each of its biometric method (noted *sn* for method *n*), as well as the performance of the fusion functions of the state of the art. We can see that biometric methods from PRIVATE have more biometric verification errors than the ones of the other databases. Using the Kruskal-Wallis test, the *sum* (p value = 0.0038) and *mul* (p value = 0.0038) operators outperformed the *min* operator. There was no significant difference (p value = 0.935) between both operators *sum* and *mul* operators.

Table 6: Performance (EER) of the Biometric Systems ($s1, s2, s3, s4$), and the State Of The Art Fusion Functions (sum, min, mul) on the Three Databases

Method		Learning	Validation
BANCA			
Biometric systems	$s1$	0.0310	0.0438
	$s2$	0.0680	0.1154
	$s3$	0.0824	0.0897
	$s4$	0.0974	0.0732
State of the art fusion	sum	0.0128	0.0128
	min	0.0385	0.0438
	mul	0.0128	0.0128
BSSR1			
Biometric systems	$s1$	0.0425	0.0430
	$s2$	0.0553	0.0620
	$s3$	0.0861	0.0841
	$s4$	0.0511	0.0454
State of the art fusion	sum	0.0116	0.0070
	min	0.0436	0.0504
	mul	0.0117	0.0070
PRIVATE			
Biometric systems	$s1$	0.1161	0.1153
	$s2$	0.1522	0.1569
	$s3$	0.0603	0.0621
	$s4$	0.2815	0.3143
State of the art fusion	sum	0.0256	0.0278
	min	0.1397	0.1471
	mul	0.0252	0.0281

5.2.2. Magnitude Of the Gain in Computation Time

We also want to prove that using the proposed EER computation method improves the computation time of the genetic algorithm run. To do that, the previously described process has been repeated two times:

- Using the proposed EER computing method with the following configuration: 5 steps and stop at a precision of 0.01.
- Using the classical EER computing method with 100 steps.

The total computation time is saved in order to compare the speed of the two systems. These tests have been done on a Pentium IV machine with 512 Mo of RAM with the Matlab programming language.

5.3. Experimental Results

5.3.1. Design of Fusion Functions

The EER of each generated function of each database is presented in Table 8 for the learning and validation sets, while Figure 10 presents there ROC

Table 7: Configurations Of The Three Weighted Functions For Each Database

BANCA	
GA	configuration
ga1	$8.7229 * s_0 + 2.3092 * s_1 + 2.0626 * s_2 + 2.9687 * s_3$
ga2	$s_0^{7.4721} * s_1^{1.8091} * s_2^{2.1255} * s_3^{0.8874}$
ga3	$-2.3079 * s_0^{-6.0105} + -8.5217 * s_1^{-3.1367} + -8.6644 * s_2^{-3.5730} + -7.0890 * s_3^{7.4634}$
BSSR1	
GA	configuration
ga1	$2.3270 * s_0 + 0.8790 * s_1 + 0.3661 * s_2 + 9.4978 * s_3$
ga2	$s_0^{2.0650} * s_1^{0.5660} * s_2^{1.6168} * s_3^{9.1864}$
ga3	$5.7285 * s_0^{4.6227} + 4.2471 * s_1^{6.8192} + 9.7541 * s_2^{6.7588} + 5.9431 * s_3^{0.9251}$
PRIVATE	
GA	configuration
ga1	$6.7755 * s_0 + 2.3841 * s_1 + 5.9128 * s_2 + 2.6919 * s_3$
ga2	$s_0^{6.2215} * s_1^{4.1538} * s_2^{6.6853} * s_3^{3.9254}$
ga3	$4.8647 * s_0^{1.6977} + 8.3564 * s_1^{5.9125} + 4.7450 * s_2^{2.2407} + 2.0707 * s_3^{0.7681}$

Table 8: EER For Training and Validation Sets And Computation Time Gain By Using Our EER Computation Method

Function	Train EER	Test EER	Gain (%)
BANCA			
(5): ga1	0.0032	0.0091	61.29
(6): ga2	0.0032	0.0091	41.84
(7): ga3	0.0037	0.0053	43
BSSR1			
(5): ga1	0.000596	0.0038	78.32
(6): ga2	0.000532	0.0038	64.77
(7): ga3	0.000626	0.0038	28.49
PRIVATE			
(5): ga1	0.019899	0.0241	77.66
(6): ga2	0.019653	0.0244	46.5
(7): ga3	0.020152	0.0217	55.03

curve on the validation set. We can see that the proposed generated functions are all globally better than the ones from the state of the art (by comparison with Table 6) and the obtained EER is always better than the ones of the *sum* and *mul*. The two new fusion functions ((6) and (7)) give similar or better results than the weighted sum (5). We do not observe over-fitting problems: the results are promising both on the learning and validation sets. We also could expect to obtain even better performance by using more individuals or more generations in the genetic algorithm process, but, in this case, timing computation would become too much important. Their will always be a tradeoff between security (biometric performance) and computation speed (genetic algorithm performance). By the way, the best individuals were provided in the

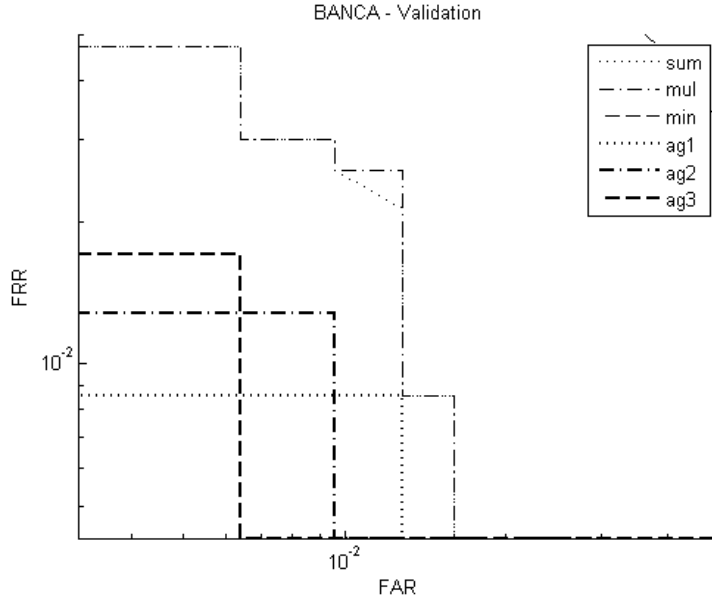


Figure 8: ROC Curve Of The Generated Multibiometrics Fusion Functions on the Validation Set of the BANCA dataset

first 10 generations, and several runs give approximately the same results, so we may already be in a global minima.

As a conclusion of this part, we increased the performance of multibiometrics systems given the state of the art by reducing errors of 58% for BANCA, 45% for BSSR1 and 22% for PRIVATE.

5.3.2. Magnitude Of the Gain in Computation Time

Table 8 presents a summary of the performance of the generated methods both in term of EER and timing computing improvement. The column gain presents the improvement of timing computation between the proposed EER polytomous computation time and the classical one in 100 steps.

We can observe that, in all the cases, the proposed computation methods outperform the classical one (which is not its slowest version). We can see that this improvement depends both on the cardinal of the set of scores and the function to evaluate: there are better improvements for (5). The best gain is about 78% while the smallest is about 28%.

5.3.3. Discussion

Once again, we can observe the interest of our EER estimation method which allows to obtain results far quickly. We can note that the generated function all generate a monotonically decreasing ROC curve which allows to use our method. If the ROC curve does not present this shape, we would be unable to

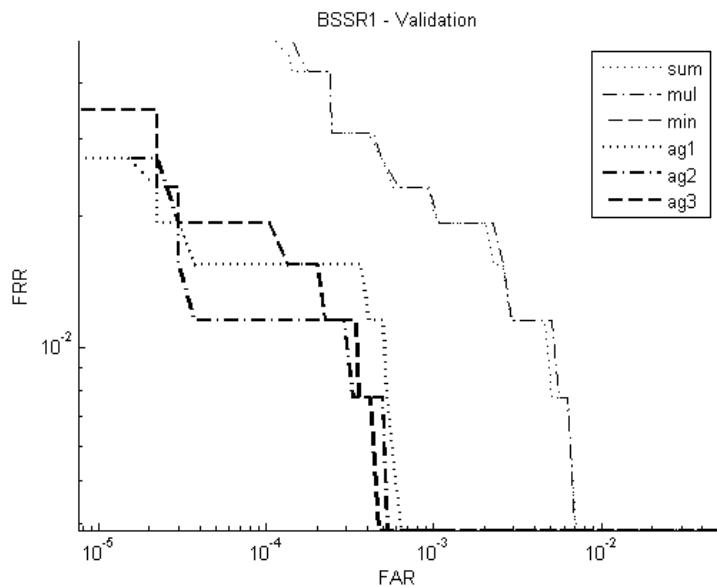


Figure 9: ROC Curve Of The Generated Multibiometrics Fusion Functions on the Validation Set of the BSSR1 dataset

obtain the estimated EER (such drawbacks as been experimented using genetic programming [41] instead of genetic algorithms).

6. Conclusion and Perspectives

The contribution of this paper is twofold: a fast approximated EER computing method (associated to its confidence interval), and two score fusion functions having to be parametrized thanks to genetic algorithms. Using these two contributions together allows to speed up the computation time of the genetic algorithm because its fitness function consists on computing the EER (thus, allows to use a bigger population).

The fast EER computing method has been validated on five different biometric systems and compared to the classical way. Experimental results showed the benefit of the proposed method, in terms of precision of the EER value and timing computation.

The score fusion functions have been validated on three significant multibiometrics databases (two reals and one chimerical) having a different number of scores. The fusion functions parametrized by genetic algorithm always outperform simple state of the art simple functions (*sum*, *min*, *mul*), and, the two new fusion functions have given better or equal results than the simple weighted sum. Using the proposed fast EER computing method also considerably speed up the timing computation of the genetic algorithms. These better results imply

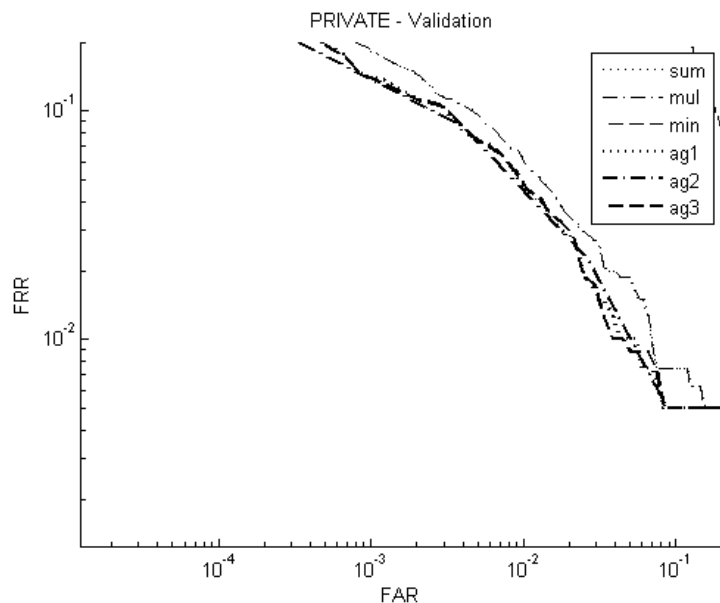


Figure 10: ROC Curve Of The Generated Multibiometrics Fusion Functions on the Validation Set of the Private dataset

that the multibiometrics system has a better security (fewer impostors can be accepted) and is more pleasant to be used (fewer genuine users can be rejected).

One limitation of the proposed method is related to the shape of the ROC curve and the attended precision wanted. In some cases, the method is unable to get the EER at the wanted precision, and, is not able to return the result (we did not encounter this case in these experiments).

Our next research will focus on the use of different evolutionary algorithms in order to generate other kind of complex functions allowing to get better results.

Acknowledgements

Work has been with financial support of the ANR ASAP project, the Lower Normandy French Region and the French research ministry.

- [1] M. Fons, F. Fons, E. Cantó, Biometrics-based consumer applications driven by reconfigurable hardware architectures, Future Generation Computer Systems In Press, Corrected Proof (2010) –.
- [2] A. Kumar, Y. Zhou, Human identification using knucklecodes, in: IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2009), 2009.

- [3] A. Ross, K. Nandakumar, A. Jain, Handbook of Multibiometrics, Springer, 2006.
- [4] A. K. Jain, S. Pankanti, S. Prabhakar, L. Hong, A. Ross, Biometrics: a grand challenge, in: Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04), 2004, pp. 935–942.
- [5] Human centred design process for interactive systems, iSO 13407:1999 (1999).
- [6] Information technology biometric performance testing and reporting, iSO/IEC 19795-1 (2006).
- [7] S. Smith, Humans in the loop: Human computer interaction and security, IEEE Security and Privacy (2003) 75–79.
- [8] L. Coventry, A. D. Angeli, G. Johnson, Biometric verification at a self service interface, in: Contemporary ergonomics, 2003.
- [9] N. Gunson, D. Marshall, F. McInnes, M. Jack, Usability evaluation of voiceprint authentication in automated telephone banking: Sentences versus digits, Interact. Comput. (2011) 57–69.
- [10] M. El-Abed, R. Giot, B. Hemery, C. Rosenberger, A study of users' acceptance and satisfaction of biometric systems, in: International Carnahan Conference on Security Technology (ICCST'10), 2010.
- [11] E. Tabassi, C. Wilson, A novel approach to fingerprint image quality, in: International Conference on Image Processing (ICIP'05), 2005, pp. 37–40.
- [12] M. El-Abed, romain giot, C. Charrier, C. Rosenberger, Evaluation of biometric systems: An svm-based quality index, in: NISK conference, 2010.
- [13] K. Kryszczuk, J. Richiardi, A. Drygajlo, Impact of combining quality measures on biometric sample matching, in: BTAS'09: Proceedings of the 3rd IEEE international conference on Biometrics: Theory, applications and systems, 2009, pp. 133–138.
- [14] N. K. Ratha, J. H. Connell, R. M. Bolle, An analysis of minutiae matching strength, in: Proceedings of the Third International Conference on Audio- and Video-Based Biometric Person Authentication, 2001, pp. 223–228.
- [15] U. Uludag, A. K. Jain, Attacks on biometric systems: A case study in fingerprints, in: Proc. SPIE-EI 2004, Security, Seganography and Watermarking of Multimedia Contents VI, 2004.
- [16] J. Galbally, R. Cappelli, A. Lumini, G. Gonzalez-de Rivera, D. Maltoni, J. Fierrez, J. Ortega-Garcia, D. Maio, An evaluation of direct attacks using fake fingers generated from ISO templates, Pattern Recogn. Lett.

- [17] Information technology – security techniques – security evaluation of biometrics, iSO/IEC FCD 19792 (2008).
- [18] M. El-Abed, R. Giot, B. Hemery, J.-J. Schwartzmann, C. Rosenberger, Towards the security evaluation of biometric authentication systems, in: IEEE International Conference on on Security Science and Technology (IC-SST'11), 2011, pp. 167–173.
- [19] N. I. of Standards, Technology, Nist biometric score set (2006).
URL <http://www.itl.nist.gov/iad/894.03/biometricscores/>
- [20] K. Nandakumar, Y. Chen, S. Dass, A. Jain, Likelihood ratio-based biometric score fusion, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2) (2008) 342.
- [21] N. Sedgwick, C. Limited, Preliminary report on development and evaluation of multi-biometric fusion using the nist bssr1 517-subject dataset, Cambridge Algorithmica Limited.
- [22] N. Poh, Banca score database, http://info.ee.surrey.ac.uk/Personal/Norman.Poh/web/banca_multi/.
- [23] A. Martinez, R. Benavente, The AR face database, Tech. rep., CVC Technical report (1998).
- [24] R. Giot, M. El-Abed, C. Rosenberger, Greyc keystroke : a benchmark for keystroke dynamics biometric systems, in: IEEE Third International Conference on Biometrics : Theory, Applications and Systems (BTAS), 2009.
- [25] F. Monrose, A. D. Rubin, Keystroke dynamics as a biometric for authentication, Future Generation Computer Systems 16 (4) (2000) 351 – 359.
- [26] R. Giot, M. El-Abed, C. Rosenberger, Biometrics, Intech, 2011, Ch. Keystroke Dynamics Overview, pp. 157–182.
URL <http://www.intechopen.com/articles/show/title/keystroke-dynamics-overview>
- [27] R. Raghavendra, B. Dorizzi, A. Rao, G. K. Hemantha, Pso versus adaboost for feature selection in multimodal biometrics, in: Biometrics: Theory, Applications, and Systems, 2009. BTAS '09. IEEE 3rd International Conference on, 2009, pp. 1–7. doi:10.1109/BTAS.2009.5339039.
- [28] A. Jain, K. Nandakumar, A. Ross, Score normalization in multimodal biometric systems, Pattern Recognition 38 (12) (2005) 2270–2285.
- [29] S. Hocquet, J.-Y. Ramel, H. Cardot, User classification for keystroke dynamics authentication, in: The Sixth International Conference on Biometrics (ICB2007), 2007, pp. 531–539.

- [30] P. Teh, A. Teoh, T. Ong, H. Neo, Statistical fusion approach on keystroke dynamics, in: Proceedings of the 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System-Volume 00, IEEE Computer Society, 2007, pp. 918–923.
- [31] F. Roli, L. Didaci, G. Marcialis, Adaptive biometric systems that can improve with use, *Advances in Biometrics*. Springer London (2008) 447–471.
- [32] J. Montalvao Filho, E. Freire, Multimodal biometric fusion – joint typist (keystroke) and speaker verification, in: Telecommunications Symposium, 2006 International, 2006, pp. 609–614.
- [33] J. Fierrez-Aguilar, J. Ortega-Garcia, D. Garcia-Romero, J. Gonzalez-Rodriguez, A comparative evaluation of fusion strategies for multimodal biometric verification, *Lecture notes in computer science* (2003) 830–837.
- [34] V. Vapnik, et al., Theory of support vector machines, Department of Computer Science, Royal Holloway, University of London (1996) 1677–1681.
- [35] R. Johnson, An introduction to the bootstrap, *Teaching Statistics* 23 (2) (2001) 49–54.
- [36] D. Seljebotn, Fast numerical computations with cython, in: Proceedings of the 8th Python in Science Conference, 2009.
- [37] G. Varoquaux. [link].
URL <http://packages.python.org/joblib/>
- [38] F. Perez, B. Granger, Ipython: a system for interactive scientific computing, *Computing in Science & Engineering* (2007) 21–29.
- [39] M. Mitchell, *An Introduction to Genetic Algorithms*, The MIT press, 1998.
- [40] J. Kittler, M. Hatef, R. P. Duin, M. Jiri, On combining classifiers, *IEEE Transactions on Pattern Analysis and Machine Intelligence and Security Informatics* 20 (3) (1998) 226–239.
- [41] R. Giot, C. Rosenberger, Genetic programming for multibiometrics, *Expert Systems With Applications*[forthcoming]. doi:10.1016/j.eswa.2011.08.066.