# Fast Learning For Multibiometrics Systems Using Genetic Algorithms

Romain Giot, Mohamad El-Abed, Christophe Rosenberger

# Fast Learning For Multibiometrics Systems Using Genetic Algorithms

Romain Giot, Mohamad El-Abed, Christophe Rosenberger
*GREYC Laboratory, ENSICAEN - Université de Caen Basse-Normandie - CNRS*
*ENSICAEN 6, boulevard maréchal Juin 14050 Caen Cedex 4 FRANCE*
{*romain.giot,melabed,christophe.rosenberger*}*@greyc.ensicaen.fr*

## ABSTRACT

*The performance (in term of error rate) of biometric systems can be improved by combining them. Multiple fusion techniques can be applied from classical logical operations to more complex ones based on score fusion. In this paper, we use a genetic algorithm to learn the parameters of different multibiometrics fusion functions. We are interested in biometric systems usable on any computer (they do not require specific material). In order to improve the speed of the learning, we defined a fitness function based on a fast Error Equal Rate computing method. Experimental results show that the developed method provides very low error rates while having reasonable computation times. The proposed method opens new perspectives for the development of secure multibiometrics systems with speeding up their computation time.*

**KEYWORDS:** Authentication, Identity Management, Access Control, Multibiometrics.

## 1. INTRODUCTION

Biometrics is a science in perpetual evolution: every day, new algorithms performing much better than previous ones are designed, as well as novel type of biometric modalities (like finger knuckle recognition proposed in 2009 [1]). Many biometric modalities exist, each classified among three main families (even if we can find a more precise topology in the literature) (i) *Biological*: recognition based on the analysis of biological data linked to an individual (e.g., DNA, EEG analysis, ...). (ii) *Behavioural*: based on the analysis of the behaviour of an individual while he is performing a specific task (e.g., signature dynamics, gait, ...). (iii) *Morphological*: based on the recognition of different physical patterns, which are, in general, permanent and unique (e.g., fingerprint, face recognition, ...).

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. Biometrics multi-modality (or multibiometrics) allows to compensate this problem while obtaining better biometric performances (i.e., better security by rejecting less genuine users and accepting less impostors) 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 biometric multi-modality. Most of them are listed here [2], they use: (a) different sensors of the same modality (i.e., capacitive or resistive sensors for fingerprint acquisition); (b) different representations of the same capture (i.e., use of points of interest or texture); (c) different biometric modalities (i.e., face and fingerprint); (d) several instances of the same modality (i.e., left and right eye for iris recognition); (e) multiple captures (i.e., 25 images per seconds in a video used for face recognition); (f) an hybrid system composed of the association of the previous ones.

In our study, we are interested in the first four kinds of multi-modality. We present in this paper, a new multibiometrics approach using fusion functions parametrized by various genetic algorithms using a fast EER (Error Equal Rate, presented in section 2.1) computing method to speed up the fitness evaluation. Our generated functions give better results than the $sum$ one which is commonly accepted as a good fusion function in the literature. The computation time needed to estimate the parameters of these functions is greatly improved by the help of our fast EER computing method.

This paper is related to high performance computing, be-

978-1-4244-6829-4/10/$26.00 ©2010 IEEE

cause algorithms are designed to work in an infrastructure managing the biometric authentication of thousands of individuals. To improve recognition rate of biometric systems, it is necessary to regularly update the model to take into account intra class variability. With our proposition, the time taken to update the model would be lower. The more the method is fast, the more we can launch the updating process.

In the next section, we present the state of the art of this domain. Section 3 describes the proposed method and section 4 illustrates its efficiency. We conclude and give some perspectives of this study.

## 2. BACKGROUND

### 2.1. Biometric Systems Evaluation

Like in all pattern recognition problems, biometric systems need to be evaluated in order to be compared together. There are three main error rates:

**FAR** *False Acceptance Rate* which represents the ratio of impostors accepted by system;

**FRR** *False Rejection Rate* which represents the ratio of genuine users rejected by the system;

**EER** *Error Equal Rate which* is the error rate when the system is configured in order to obtain a FAR equal to the FRR. This is the error rate we use all over this paper.

The main approach to compute the EER is to compute the couples of FAR and FRR for several thresholds. These thresholds are linearly distributed over the minimal and the maximal scores of the whole set of intra and inter scores. Each score is computed by comparing a capture to a model (this model can be a reference capture or computed with several captures). The intra scores are computed by comparing captures from one user with its own model, while the inter scores are computed by comparing captures from one user with model from another one. This approach is slightly different than in [3], but, when the number of scores is much more important than the number of steps (which is always the case in our biometric databases), the number of comparison is far lower (and takes less computation time).

The computation of the FAR and FRR 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.

$$FRR = \frac{Card\{intra_i, \forall_{i \subset Card(intra)} | intra_i > thr\}}{Card\{intra\}} \tag{1}$$

$$FAR = \frac{Card\{inter_i, \forall_{i \subset Card\{inter\}} | inter_i \leq thr\}}{Card\{inter\}} \tag{2}$$

With the help of these equations, we can see that it is necessary to operate $Card\{intra\} + Card\{inter\}$ comparisons for each couple of (FAR, FRR), so the computation time for these comparisons is highly related to the number of available scores. Once the couples of (FAR, FRR) are computed for each threshold, we can obtain:

- *the ROC curve* by plotting the FRR depending on the FAR (we choose this representation to easily graphically read the EER which is presented below). 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.
- *the EER* by selecting the couple of (FAR, FRR) having the smallest absolute difference (3) and returning their average (4). By this way, we have obtained the best approaching EER with the smallest precision error. The EER computing algorithm is presented in the Figure 1.

$$ERROR_i = min(abs(FAR_i - FRR_i)), \quad \forall_{i \subset Card\{ROC\}} \tag{3}$$

$$EER = \frac{FAR_i + FRR_i}{2}, \quad \forall_{i \subset Card\{ROC\}} | i = argmin(ERROR_i) \tag{4}$$

### 2.2. 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 fusion process is the most important process in multimodal systems. It can be operated on the scores provided by algorithms or in the templates themselves [4]. In the first case, it is necessary to normalize the different scores as they do not evolve in

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

**Figure 1. Classical EER Computing Algorithm**

the same range. Different methods can be used for doing this, and the more efficient methods are *zscore*, *tanh* and *minmax* [5]. 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 [6], three different keystroke dynamics implementations are fused with an improvement of the EER, but less than 40 users are involved in the database. In [7], 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 [8], authors use both face and fingerprint recognition, the impact of error rate reduction is used to reduce the error when adapting the user's model. There is only one paper (to our knowledge) on keystroke dynamics fusion with another kind of biometric modality (voice recognition): it is presented in [9], but only 10 users are involved in the experiment. In [10], multi-modality is done on fingerprints, speech, and face images on 50 individuals. Fusion has been done with SVM [11] with good improvements, especially, when using user specific classifiers.

Very few multimodal systems have been proposed for 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. PROPOSED METHOD

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.

## 3.1. Fast EER Computing

Computation time to get the EER can be quite important. When EER need to be computed a lot of time, it is necessary to use a quicker 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 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 EER, we need to find the $x_{DIFF}$ for which $y_{DIFF}$ is closest as possible to zero. An analogy with the classical EER computing, would be to incrementally compute $y_{DIFF}$ for each score by increasing order and stop when $y_{DIFF}$ change of sign. By this way, we can expect to do half threshold configuration 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 $O(log(n))$. We have implemented a polytomous version of EER computing:

1. we chose $i$ thresholds linearly distributed on the scores sets
2. for each scores, we compute the FAR and FRR
3. we catch the two following scores $s1$ and $s2$ having $sign(FRR_{s1} - FAR_{s1})$ different of $sign(FRR_{s2} - FAR_{s1})$
4. we repeat step 2 with selecting $i$ thresholds between $s1$ and $s2$ included while $FRR_{s1} - FAR_{s1})$ do 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 $i$. A tentative of analysis of the method is presented in the next section.

The algorithm is given in Figure 2, while its working steps are given in Figure 3. It is applied on a system which ROC curve is plotted with the plain line. We have chosen $i = 4$ points to compute during each iteration. For the first iteration, 4 points are computed (represented by a circle). Then, according to the sign change of the difference, 2 points are chosen and the steps are repeated: two more points (the interval extremities were already computed) are computed;

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

**Figure 2. Fast EER Computing Algorithm**

they are represented by a star. The same phenomena appears in the iteration 3, where points are represented by a rectangle. These tests are repeated until the difference between FAR and FRR of a point is lower than the required precision. When it is verified, the EER is estimated by computing the mean of the FAR and FRR of this point. The ROC curve generated by the polytomous algorithm is represented in the dotted line. We can see that the inevitable trade-off (due to this computing time earning) is the lack of precision of the ROC curve. But, in our case, it is not a problem, because our algorithm is not designed for that purpose.

### 3.2. Fusion Method

We have tested three different kinds of score fusion methods which parameters are automatically set by genetic algorithms [12]. 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 weight of multiplication 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 bigger scores.

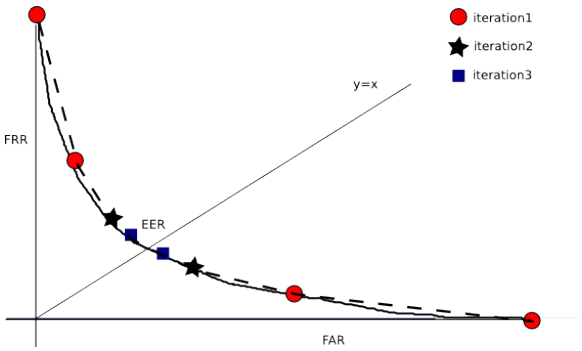



**Figure 3. Points Computed By Our Algorithm When $i = 4$. In This Case, The EER Is Found In 8 Comparisons**

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

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

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

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 $w_i$ 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) is 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.

## 4. VALIDATION

We have designed different experimental protocols in order to answer to several questions:

- How fast and better is our EER computing method in comparison to the classical one ?
- Do our fusion functions provide better results than the fusion functions from the state of the art ?

- What is the gain of using our EER method for the learning step of our multibiometrics system ?

## 4.1. Databases Sets

In order to do these evaluations, we have used three different biometric databases:

#### 4.1.1. BSSR1

The BSSR1 [13] 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 [14, 15].

#### 4.1.2. BANCA

The second database is a subset of scores produced from the BANCA database [16]. 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

G1 set is used as the learning set, while G2 set is used as the validation set.

#### 4.1.3. PRIVATE

The latest database is a chimeric one we have created for this purposed by combining two public biometric template databases: the AR [17] for the facial recognition and the GREYC keystroke [18] for keystroke dynamics. The AR database is composed of frontal facial images of 126 individuals under different facial expression, 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. User were asked to type the password "greyc laboratory" 6 times on a laptop and 6 times on an USB keyboard by interlacing the typings. 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 model of each user and the 16 other 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.

## 4.2. Evaluation of the EER Computing

#### 4.2.1. Experimental Protocol

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* our 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 our method is better than the classical one, and what are its best parameters.

#### 4.2.2. Experimental Results

Table 1 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 (8), 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.

$$T = n * (A * (Card\{intra\} + Card\{inter\}) + B) + C \quad (8)$$

We can see that the computation time is highly related to the number of comparisons and the size of the score set. 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 values, we can say that the fastest algorithms are the one having the smallest number of tests.

Based on the number of comparisons (and by the way, the

**Table 1. Comparison Of The Differents EER Computing Methods And Configurations On The First Test Set**

| 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 2. 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 |

timing computation), Table 2 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 our 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 our EER computation will speed up genetic algorithms using the EER as fitness function.

## 4.3. Efficiency of Fusion Functions

### 4.3.1. Experimental Protocol

Table 3 presents the parameters of the genetic algorithms. The genetic algorithms has been trained on a learning set composed of half of the intra-scores and half of the inter-scores of a database and they have been verified with a validation set composed of the others scores. The three

**Table 3. 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 gemetric selection (probability of 0.9) |
| Mutation | boundary, multi non uniform, non uniform, uniform |
| Cross-over | Heuristic Crossover |
| Elitism | True |

databases have been used separately.

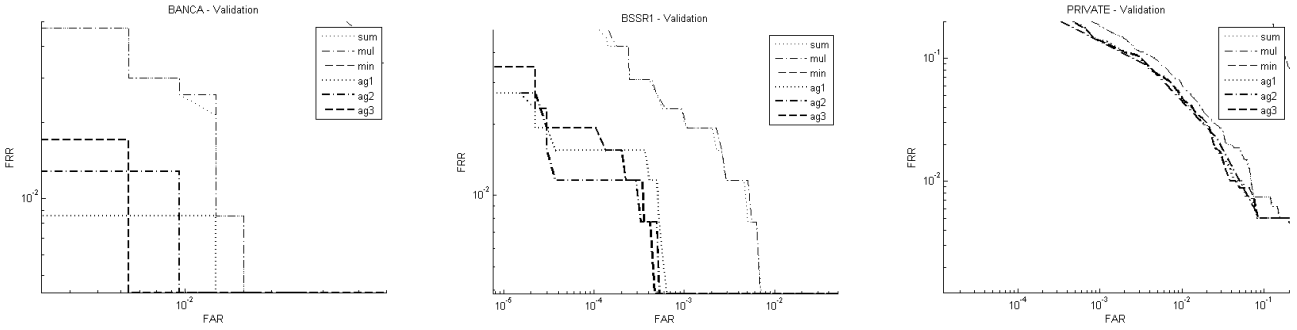
The generated functions are compared to three methods of the state of the art: $sum$, $mul$ and $min$, they have been explored in [5, 19]. Table 4 presents, for each database, the EER of each of its biometric method (noted $sn$ for method $n$), as well as the performance of the fusion functions of the state of the art.

We can see that biometric methods from PRIVATE have more biometric verification errors than the ones of the other databases. The $min$ fusion operator does not give good results, while $sum$ and $mul$ operators always improve the performance of the system.

### 4.3.2. Experimental Results

The EER of each generated function of each database is presented in Table 5 for the learning and validation sets, while Figure 4 presents there ROC curve on the validation set. We can see that our generated functions are all globally better than the ones from the state of the art (by comparison with Table 4) 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




(a) Performance With BANCA (b) Performance With BSSR1 (c) Performance With PRIVATE

**Figure 4. ROC Curve Of The Generated Multibiometrics Fusion Functions On The Validation Set**

**Table 4. 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 |

the way the best individuals were provided in the 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.

**Table 5. 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 |

## 4.4. Magnitude Of the Gain in Computation Time

### 4.4.1. Experimental Protocol

We also want to prove that using our 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 our 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.

### 4.4.1. Experimental Results

Table 5 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 our EER polytomous computation time and the classical one in 100 steps.

We can observe that, in all the cases, our 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. CONCLUSION

In this paper, we have presented two things: a fast EER computing method, and two fusion functions having to be parametrized thanks to genetic algorithms. The benefit of this fast EER computing method is to speed up the computation time of the genetic algorithm because its fitness function consists on computing the EER.

The fast EER computing method have been validated on five different biometric systems and compared to the classical way. The results present the superiority of our method, both in term of precision of the EER value and timing computation.

The score fusion functions have been validated on three significant multibiometric databases (two reals and one chimerical). The fusion functions parametrize by genetic algorithm always outperform simple state of the art simple functions ($sum$, $min$, $mul$), and the two new fusion functions have given better are equal results than the weighted sum. Using our fast EER computing method also considerably speed up the timing computation of the genetic algorithms. These better results imply 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).

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. EER computation timing may also be improved by using the fixed point theorem.

## REFERENCES


[1] A. Kumar and Y. Zhou, "Human Identification Using KnuckleCodes," in *IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2009)*, 2009.

[2] A. Ross, K. Nandakumar, and A. Jain, *HANDBOOK OF MULTIBIOMETRICS*. Springer, 2006.

[3] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[4] R. Raghavendra, B. Dorizzi, A. Rao, and 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*, Sept. 2009, pp. 1–7.

[5] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," *Pattern Recognition*, vol. 38, no. 12, pp. 2270–2285, 2005.

[6] S. Hocquet, J.-Y. Ramel, and H. Cardot, "User classification for keystroke dynamics authentication," in *The Sixth International Conference on Biometrics (ICB2007)*, 2007, pp. 531–539.

[7] P. Teh, A. Teoh, T. Ong, and 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.

[8] F. Roli, L. Didaci, and G. Marcialis, "Adaptive biometric systems that can improve with use," *Advances in Biometrics. Springer London*, pp. 447–471, 2008.

[9] J. Montalvao Filho and E. Freire, "Multimodal biometric fusion – joint typist (keystroke) and speaker verification," in *Telecommunications Symposium, 2006 International*, 2006, pp. 609–614.

[10] J. Fierrez-Aguilar, J. Ortega-Garcia, D. Garcia-Romero, and J. Gonzalez-Rodriguez, "A comparative evaluation of fusion strategies for multimodal biometric verification," *Lecture notes in computer science*, pp. 830–837, 2003.

[11] V. Vapnik *et al.*, "Theory of support vector machines," *Department of Computer Science, Royal Holloway, University of London*, pp. 1677–1681, 1996.

[12] M. Mitchell, *AN INTRODUCTION TO GENETIC ALGORITHMS*. The MIT press, 1998.

[13] N. I. of Standards and Technology, "Nist biometric score set," 2006. [Online]. Available: http://www.itl.nist.gov/iad/894.03/biometricscores/

[14] K. Nandakumar, Y. Chen, S. Dass, and A. Jain, "Likelihood ratio-based biometric score fusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, p. 342, 2008.

[15] N. Sedgwick and C. Limited, "Preliminary Report on Development and Evaluation of Multi-Biometric Fusion using the NIST BSSR1 517-Subject Dataset," *Cambridge Algorithmica Linited*, 2005.

[16] N. Poh, "Banca score database," http://info.ee.surrey.ac.uk/Personal/Norman.Poh/web/banca_multi/.

[17] A. Martinez and R. Benavente, "The ar face database," CVC Technical report, Tech. Rep., 1998.

[18] R. Giot, M. El-Abed, and R. Christophe, "Greyc keystroke: a benchmark for keystroke dynamics biometric systems," in *IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2009)*, 2009.

[19] J. Kittler, M. Hatef, R. P. Duin, and M. Jiri, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence and Security Informatics*, vol. 20, no. 3, pp. 226–239, march 1998.