HAL
open science

Genetic Programming for Multibiometrics

Romain Giot, Christophe Rosenberger

To cite this version:

Romain Giot, Christophe Rosenberger. Genetic Programming for Multibiometrics. Expert Systems with Applications, 2012, 39 (2), pp.1837–1847. 10.1016/j.eswa.2011.08.066 . hal-00671952

HAL Id: hal-00671952

https://hal.science/hal-00671952v1

Submitted on 20 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.

Genetic Programming for Multibiometrics

Romain Giot*, Christophe Rosenberger

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

Abstract

Biometric systems suffer from some drawbacks: a biometric system can provide in general good performances except with some individuals as its performance depends highly on the quality of the capture... One solution to solve some of these problems is to use multibiometrics where different biometric systems are combined together (multiple captures of the same biometric modality, multiple feature extraction algorithms, multiple biometric modalities...). In this paper, we are interested in score level fusion functions application (i.e., we use a multi-biometric authentication scheme which accept or deny the claimant for using an application). In the state of the art, the weighted sum of scores (which is a linear classifier) and the use of an SVM (which is a non linear classifier) provided by different biometric systems provid one of the best performances. We present a new method based on the use of genetic programming giving similar or better performances (depending on the complexity of the database). We derive a score fusion function by assembling some classical primitives functions ($+$, $*$, $-$, ...). We have validated the proposed method on three significant biometric benchmark datasets from the state of the art.

Keywords: Multibiometrics, Genetic Programming, Score fusion, Authentication.

*Corresponding author
Email addresses: romain.giot@ensicaen.fr (Romain Giot), christophe.rosenberger@greyc.ensicaen.fr (Christophe Rosenberger)

1. Introduction

1.1. Objective

Every day, new evolutions are brought in the biometric field of research. These evolutions include the proposition of new algorithms with better performances, new approaches (cancelable biometrics, soft biometrics, ...) and even new biometric modalities (like finger knuckle recognition [1], for example). There are many different biometric modalites, each classified among three main families (even if we can find a more precise topology in the literature) :

- *biological* : recognition based on the analysis of biological data linked to an individual (e.g., DNA analysis [2], the odor [3], the analysis of the blood of different physiological signals, as well as heart beat or EEG [4]);
- *behavioural* : based on the analysis of an individual behaviour while he is performing a specific task (e.g., keystroke dynamics [5], online handwritten signature [6], the way of using the mouse of the computer [7], voice recognition [8], gait dynamics (way of walking) [9] or way of driving [10]);
- *morphological* based on the recognition of different particular physical patterns, which are, for most people, permanent and unique (e.g., face recognition [11], fingerprint recognition [12], hand shape recognition [13], or blood vessel [14], ...).

Nevertheless, there will always be some users for which a biometric modality (or method applied to this modality) gives bad results, whereas, they are better in average. These low performances can be implied by different facts: the quality of the capture, the instant of acquisition and the individual itself but they have the same implication (impostors can be accepted or user need to authenticate themselves several times on the system before being accepted). Multibiometrics allow to solve this problem while obtaining better performances (i.e., better security by accepting less impostors and better user acceptance by rejecting less genuine users) and by expecting that errors of the different modalities are not

correlated. In this paper, we propose a generic approach for multibiometric systems.

We can find different types of biometric multimodalites [15]. They use:

1. different sensors of the same biometric modality (i.e., capacitive or resistive sensors for fingerprint acquisition);
2. several different representations for the same capture (i.e., use of points of interest or texture for face or fingerprint recognition);
3. different biometric modalities (i.e., face and fingerprint recognition);
4. different 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.

We are interested in the first four cases in this paper. Our objective is to automatically generate fusion functions which combine the scores provided by different biometric systems in order to obtain the most efficient multibiometrics authentication scheme.

1.2. Background

1.2.1. Performance Evaluation

In order to compare different multibiometrics systems, we need to present the how to evaluate them. Several works have already done on the evaluation of biometric systems [16, 17]. Evaluation is generally realized within three aspects:

- *performance*: it has for objective to measure various statistical criteria on the performance of the system (*Capacity* [18], *EER*, *Failure To Enroll (FTE)*, *Failure To Acquire (FTA)*, computation time, *ROC* curves, etc [17]);
- *acceptability*: it gives some information on the individuals' *perception*, *opinions* and *acceptance* regarding the system;

- *security*: it quantifies how well a biometric system (algorithms and devices) can resist to several types of logical and physical attacks such as Denial of Service (DoS) attack.

In this paper, we are only interested in performance evaluation (because the fusion approach is not modality dependant and perception and security depend on the used modalities). The main performance metrics are the following ones:

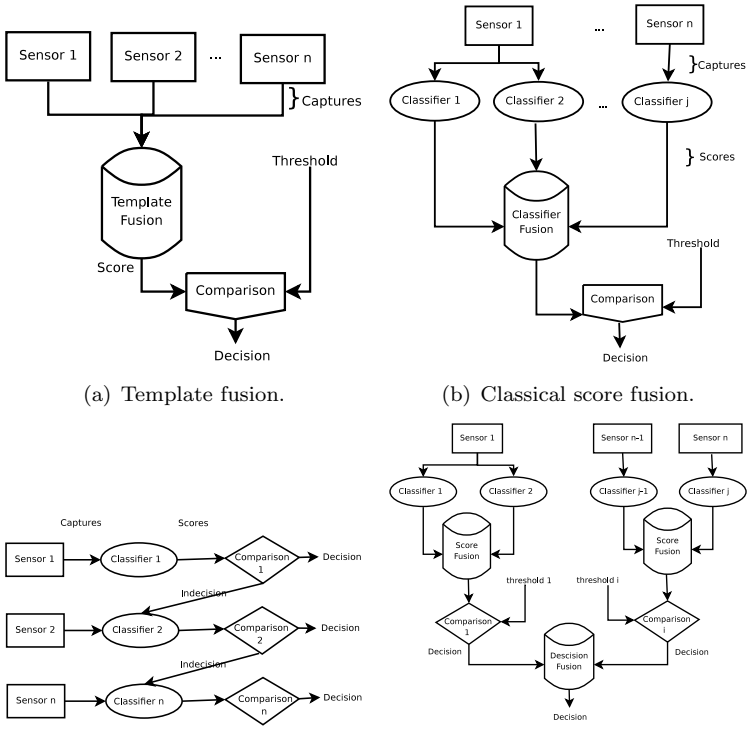
- *FAR (False Acceptance Rate)* which represents the ratio of impostors accepted by the system;
- *FRR (False Rejection Rate)* which represents the ratio of genuine users rejected by the system;
- *EER (Error Equal Rate)* which is the error rate when the system is configured in order to obtain a *FAR* equal to the *FRR*;
- *ROC (Receiver Operating Characteristic)* curve which plots the *FRR* depending on the *FAR* and gives an overall overview of system performance;
- *AUC (Area Under the Curve)* which gives the area under the ROC curve. In our case, smaller is better. It is a way to globally compare performance of different biometric systems.

We can also present the *HTER* (Half Total Error Rate) which is the mean between the *FAR* and *FRR* for a given threshold (this error rate is interesting when we cannot get the *EER*).

1.2.2. Biometric Fusion

There are several studies on multibiometrics. The fusion can be operated on different points of the mechanism:

- *template fusion*: the templates captured by different biometric systems are merged together, then the learning process is realized on these new templates [19, 20]. Figure 1(a) presents this type of fusion. The fusion



(a) Template fusion.

(b) Classical score fusion.

(c) Cascade fusion.

(d) Hierarchical fusion.

Figure 1: Illustration of different fusion mechanisms.

process is related to a feature selection in order to determine the most significant patterns to minimize errors.

- *decision fusion*: the decision is taken for each of the biometric authentication system, then the final decision is done by fusing the previous ones [21].
- *rank fusion*: the decision is done with the help of different ranks of biometric identification systems. The main method is the majority vote [22].
- *score fusion*: the fusion is realized considering the output of the classifiers. The Figure 1(b) presents this type of fusion.

Buyssens *et al.* [23] showed the interest of biometric fusion for face recognition combining the image in visible and infrared color spaces with convolutional

neural networks. In [24], Mantalvao and Freire have combined keystroke dynamics with voice recognition, it seems it is the first time that multibiometrics has been done with keystroke dynamics and another biometric modality. In [25], Hocquet *et al.* demonstrated the interest of fusion in keystroke dynamics in order to improve the recognition rates: three different keystroke dynamics functions are used on the same capture. The sum operator (consisting in summing the different scores) seems to be the most powerful approach in the literature.

These fusion architectures are quite simple but powerful. Results can yet be improved (in term of error rate or computation time) by using different architectures. A cascade fusion [26] is another interesting approach. A first test is done, if the user is correctly verified as the attended client or if it is detected as an impostor, the algorithm stops. Otherwise, another biometric authentication (with another capture from another modality) is proceeded until obtaining a decision of acceptance or rejection, or reaching the end of the cascade. So, instead of using one decision threshold, each test (except the last one) needs two thresholds: one for rejection and one for acceptance. All scores between these thresholds are considered in an indecision zone. This mechanism is presented in Figure 1(c). Another advantage of this method is to decrease the verification time by not using all the modalities, they are used only if necessary. This method has been successfully applied on a multibiometric system using face and fingerprint recognition in a mobile environment (where acquisition and computation times are important) [26].

Another kind of architecture has been proposed: it is a hierarchical fusion scheme [27] (called multiple layers by their authors). Shen *et al.* have presented this method with two different keystroke dynamics methods. The fusion is done at different steps, and involves different mathematical operations on scores (sum, weighted sum, product, min, max) and logical operations decision (comparison to a threshold, or, and) on differents templates extracted from the same capture. An extended version to any multibiometric system is presented in Figure 1(d). We think our work can be seen as a generalization of this paper.

It is also possible to model the distribution of the genuine and impostor matching scores, we talk about *Density-based score fusion*. In [28], scores are modelled with a Gaussian Mixture Model and have been tested on three multi-biometric databases involving face, fingerprint, iris and speech modalities.

Concerning non linear algorithms, Support Vector Machine (SVM) can also be used in a fusion process. Each score to combine is arranged in a vector and a training set is used to learn the SVM model. In [29], the SVM fusion to improve face recognition gives slightly better performances than weighted sum. Voice and online signature have been fused with SVM in [30]. In this experiment, arithmetic mean gives best results with noise free data, while SVM gives equivalent results with noisy data.

1.3. Discussion

In this paper, we are interested in biometric modality independent *transformation-based score fusion* [28] where the matching scores are first normalized and second combined. We have previously seen that in this case, arbitrary functions are often used. Our work is based on these various fusion architectures based on score fusion in order to produce a score fusion function automatically generated with genetic programming [31].

By the way, the definition of a fusion architecture is still an open issue in the multibiometrics research field [32], because the range of possible fusion configurations is very large. We think that using automatically generated fusion functions can bring a new solution to solve this kind of problems.

2. Material and Methods

In this section, we present all the required information in order to allow other researchers to reproduce our experiment.

2.1. Biometric databases

As it is well known that results can be highly related to the database, for this study, we have used three different multibiometric databases: the first one is the BSSR1 [33] distributed by the NIST [34] (referenced as BSSR1 in the paper), the second one is a database we have created for this purpose (referenced as PRIVATE in the paper) and the third one is a subset of scores computed with the BANCA [35] database (referenced as BANCA in the text. In fact, BANCA database is composed of templates. We have used the scores available in [36]). As all these databases are multi-modal, the scores are presented with tuples: the ith tuple of scores is represented as $s_i = (s_i^1, s_i^2, ..., s_i^n)$ for a database having n modalities (in our case, $n \in \{4, 5\}$).

The three databases are presented in detail in the following subsections while Table 1 presents a summary of their description.

2.1.1. BSSR1 database

The BSSR1 [33] database consists of 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. We have 512 tuples of intra-scores (comparison of the capture of an individual with its model) and $512 * 511 = 261,632$ tuples of inter-scores (comparison of the capture of an individual with the model of another individual). Each tuple is composed of 4 scores: $s = (s_{bssr1}^1, s_{bssr1}^2, s_{bssr1}^3, s_{bssr1}^4)$, they respectively represent the score of the algorithm A of face recognition, the score of algorithm B of face recognition (the same face image is used for the two algorithms), the score of the fingerprint recognition with left index, the score of fingerprint recognition with right index. This database has been used several times in the literature [28, 37].

2.1.2. PRIVATE database

The second database is a chimeric one we have created by combining two public biometric template databases: the AR [38] for the facial recognition and

the GREYC keystroke [39] for keystroke dynamics.

The AR database is composed of frontal facial images of 126 individuals under different facial expression, illumination conditions or occlusions. This is a quite difficult database in reason of these specificities. These images have been taken during two different sessions with 13 captures per session. The GREYC keystroke contains the captures on several session during a two months period involving 133 individuals. Users were asked to type the password "greyc laboratory" 6 times on a laptop and 6 times on an USB keyboard by interleaving 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 remaining ones to compute the intra and inter scores.

These scores have been computed by using two different methods for the face recognition (the scores $s^1_{private}$ and $s^2_{private}$ and three different ones for the keystroke dynamics ($s^3_{private}$, $s^4_{private}$ and $s^5_{private}$ scores). The face recognition algorithms are based on eigenfaces [11] and SIFT keypoints [40] comparisons between images from the model and the capture [41]. Keystroke dynamics scores have been computed by using different methods [42] based on SVM, statistical information and rhythm measures.

2.1.3. BANCA database

The lastest used benchmark is a subset of scores produced by the help of the BANCA database [36]. The selected scores correspond to the following one labelled: IDIAP_voice_gmm_auto_scale_25_100_pca.scores for s^1_{banca}, SURREY_face_nc_man_scale_100.scores for s^2_{banca}, SURREY_face_svm_man_scale_0.13.scores for s^3_{banca} and UC3M_voice_gmm_auto_scale_10_100.scores for s^4_{banca}.

We have empirically chosen this subset. G1 set is used as the learning set,

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

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

while G2 set is used as the validation set. Users from G1 are different than users from G2.

2.1.4. Discussion

The main differences between these three benchmarks are:

- the biometric modalities used in BSSR1 and BANCA have better performances than the ones in PRIVATE;
- the quantity of intra-scores is more important in PRIVATE (only one tuple of intra-score per user in BSSR1 instead of several in PRIVATE);
- BSSR1 and BANCA are databases of scores (by the way, we do not know the biometric systems having generated them) whereas PRIVATE is a database of templates (we had to compute the scores);
- BSSR1 and BANCA are more adapted to physical access control applications (i.e., a building is protected by a multi-modal biometric system), while PRIVATE is more adapted to logical access control (i.e., the authentication to a Web service is protected by a multi-modal biometric system).

In the following subsections, we describe the proposed methodology to automatically generate a score fusion function with genetic programming. We adopt the classical score fusion context described in Figure 1(b). Before using the scores provided by different biometric systems, we need to normalize them.

2.2. Score Normalization

It is necessary to normalize the various scores before operating the fusion process: indeed, these scores come from different classifiers and their values do not

necessarily evolve within the same interval. We have chosen to use the *tanh* [43] operator to normalize the scores of each modality. Equation (1) presents the normalization method, where μ^m_{gen} and σ^m_{gen} respectively represents the average and standard deviation of the genuine scores of the modality m. The genuine scores are obtained by comparing the model and the capture of the same user: they are also called the *intra scores*. In opposition, the *inter scores* are obtained by comparing the model of a user with the capture of other users. $score'$ and $score$ respectively represents the scores after and before normalisation.

$$score' = \frac{1}{2}\left\{tanh\left(\frac{1}{100}(\frac{score - \mu^m_{gen}}{\sigma^m_{gen}}\right) + 1\right\} \tag{1}$$

We have selected this normalization procedure from the state of the art because it is known to be stable [44] and does not use impostors patterns which can be hard or impossible to obtain in a real application. The aim of this paper is not to analyse the performance of biometric systems depending on the normalization procedure, but to present a new multibiometrics fusion procedure. The scores of each modality have been normalized using this procedure.

2.3. Fusion Procedure

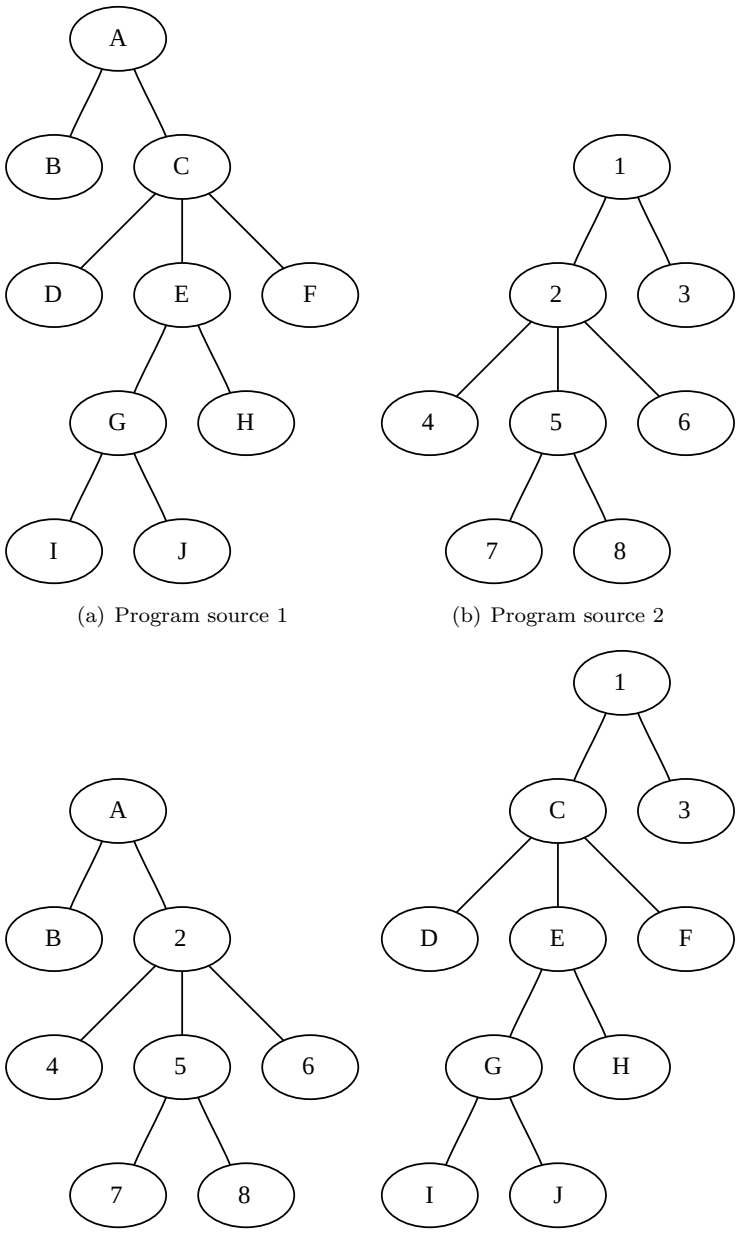
In this study, we have chosen to use genetic programming [31] in order to generate score fusion functions. Genetic programming belongs to the family of evolutionary algorithms and its scheme is quite similar to the one of genetic algorithms [45]: a population of computer programs (possibly represented by a tree) evolves during several generations; different genetic operators are used to create the new population. Programs are evaluated by using a fitness function which produces a value that is used for their comparisons and gives a probability of selection during the tournaments. In a system where the computer programs are represented by trees, their leaves mainly represent the entries of the problem, the root gives the solution to the problem and the other nodes are the various functions taking into arguments the values of their children nodes.

The leaves are called terminals and can be of several kinds: (a) pseudo-variables containing the real entries of the problem (in our case, the list of

scores of each modality), (b) some constants possibly randomly generated, (c) functions without any arguments having any side effect, or (d) some ordinary variables.

The different genetic operators usually used during the evolution are (a) the crossover, where randomly choose sub-trees have two different trees are exchanged, (b) the mutation, where a sub-tree is destroyed and replaced by another one randomly generated, or (c) the copy, where the tree is conserved in the next generation. The different steps of a genetic programming engine are presented as following:

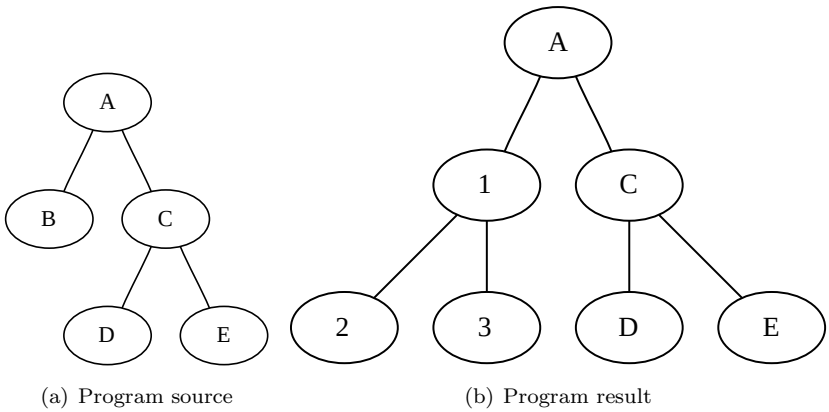
1. An initial population is randomly generated. This population is composed of computer programs using the available functions and terminals. The trees are built using a recursive procedure.
2. The following steps are repeated until the termination criterion is satisfied (the fitness function has reached the right value, or we reached the maximum number of generations).
   (a) Computation of the fitness measure of each program (the programming is evaluated according to its input data).
   (b) Selection of programs with a probability based on their fitness to apply them the genetic operations.
   (c) Creation of the new generation of programs by applying the following genetic operations (depending on their probabilities) to the previously selected programs:
      - Reproduction: the individual is copied to the new population.
      - Crossover: A new offspring program is created by recombining randomly chosen parts from two select programs. An example is provided in Figure 2.
      - Mutation: A new offspring program is created by mutating one node of the selected program at a randomly chosen place. An example is provided in Figure 3.
3. the single best program of the whole population is designated as the winner. This can be the solution or an approximate solution to the problem.



(a) Program source 1 (b) Program source 2

(c) Program result 1 (d) Program result 2

Figure 2: Crossover in genetic programming: node C from tree 1 is exchanged with node 2 from tree 2. Program result 1 is the new individual to add to the new generation.



(a) Program source (b) Program result

Figure 3: Mutation in genetic programming: node B is replaced by another sub-tree.

Different applications to genetic programming are presented in [46] as well as their bibliographic references. The fields of these applications can be listed in curve fitting, data modelling, symbolic regression, image and signal processing, economics, industrial process control, medicine, biology, bioinformatics, compression... but, it seems, so far of our knowledge, that it has not been yet applied to multibiometrics. We only found one reference on genetic programming in the biometrics field. In this paper [47], authors have used genetic programming to learn speaker recognition programs. They have used an island model where different islands operate their genetic programming evolution, and, after each generation some individuals are able to leave to another island. The obtained performance was similar to the state of the art in speaker recognition in normal conditions, but, the generated systems performed better in degraded conditions.

More information about the configuration of the genetic programming system is presented in the next section.

2.4. Parameters of the Genetic Programming

We want to use a score fusion function that returns a score related to the performance of a multibiometric system. This score has to be compared with a threshold in order to make the decision of acceptance or rejection of the user.

In this case, none logical operation is required in the generated programs and different information can be extracted from the result of the fusion function (we can compute the ROC curve, the EER, ...).

2.4.1. Fitness Function

The EER (Error Equal Rate) is usually used to compare the performance of different biometric systems together. A low EER means that FAR and FRR are both low and the system has a good performance if its threshold is configured accordingly to obtain this value. For this reason, we have chosen to use this running point to evaluate the performance of the generated score fusion functions.

To compute the EER, we consider the highest and lowest values in the final scores generated by the genetic programming. Then, we set a threshold at the lowest score and linearly increment it until obtaining the highest score value in 1000 steps. For each of these steps, we compute the FAR (comparison between the threshold and the inter scores) and FRR (comparison between the threshold and the intra scores). The ROC curve can be obtained by plotting all these couples of (FAR, FRR), while the EER is the mean of FAR and FRR for the couple having the lowest absolute difference. So, the fitness function is $fitness = (FAR_i + FRR_i)/2$, where i is the threshold for which $\text{abs}(FAR_i - FRR_i)$ is minimal.

2.4.2. Genetic Programming Parameters

In this section, we present the various parameters used in the genetic programming algorithm. Table 2 presents the various parameters of the evolutionary algorithm.

To achieve this experiment, we used the PySTEP [48] library. The generated programs contain basic functions ($+$, $-$, $*$, $/$, min, max, avg). The terminals are the scores of the biometric systems and random constants between 0 and 1.

The whole fitness cases are completed with a single tree evaluation, thanks to the numpy [49] library. Each fitness case is a tuple of scores (where each score

Table 2: Summary of the configuration of the genetic programming iterations. Numbers used in function set can be scores or constants.

Configuration	**Values**
Objective	Generates a function producing a multibiometrics score.
Functions set	• $+$: addition of two numbers, • $-$: subtraction of two numbers, • $*$: multiplication of two numbers, • $/$: division of two numbers, • min: returns the minimum of two numbers, • max: returns the maximum of two numbers, • avg: returns the mean of two numbers
Fitness function	Computes the EER of the multibiometric system
Terminal set	**BSSR1** • a: scores from s^1_{bssr1}, • b: scores from s^2_{bssr1}, • c: scores from s^3_{bssr1}, • d: scores from s^4_{bssr1}, • 50 constants linearly distributed between 0 and 1. **PRIVATE** • a, b, c: keystroke dynamics scores ($s^3_{private}$, $s^4_{private}$, $s^5_{private}$), • d, e: face recognition scores ($s^1_{private}$, $s^2_{private}$), • 50 constants linearly distributed between 0 and 1. **BANCA** • a: scores from s^1_{banca}, • b: scores from s^2_{banca}, • c: scores from s^3_{banca}, • d: scores from s^4_{banca}, • 50 constants linearly distributed between 0 and 1.
Initial population	500 random trees with a depth between 2 and 8 built with the ramped half and half method.
Evolution parameters	• Number of individuals: 500, • Maximal number of generations: 50, • Depth limited to: 8, • Probability of crossover: 45%, • Probability of mutation: 50% • Probability of reproduction: 5% (with elitism), • Selection: tournament of size 10 with a selection probability of 80%.
Termination criterion	Best individual has a fitness inferior at 0.001 (by the way, this value would never be met ...) or maximal number of generations reached.
Learning set	First half of the intra-scores tuples and first half of the inter-scores tuples.
Validating set	Second half of the intra-scores tuples and second half of the inter-scores tuples.

comes from a different biometric modality) and its result value is the score returned by the generated multimodal system. The global fitness value of a tree is the EER value computed with the previously generated scores (computation of the ROC curve, then reading of the EER value from it).

PySTEP is a strongly typed genetic programming engine, but, in our case, we do not use any particular constraints: the root node can only have a function as child (no terminal in order to avoid an unimodal system, and any function of the set), while the other function nodes can have any of the functions as children as well as any of the terminals.

The maximal depth of the generated trees is set to 8. In order to avoid to stay in a local minimal solution, the mutation probability is set to 50%. 500 individuals evolve during 50 generations. We have set this few quantities, because during our investigations, using a population of 5000 individuals on 100 generations did not give so much better results (gain not interesting in comparison to the computation time). Each database has been splitted in two sets of equal size: the first half is the learning set and the second half is the validation set.

The mutation rate is set to 50%, the cross-over rate to 45% and the reproduction rate to 5%. For mutation and cross-over the individuals are selected with a tournament of size 10 with a probability of 80% to select the best individual. The same individual can be selected several times. For the reproduction, the individuals are selected with an elitism scheme: the 5% best individuals are copied from generation $n-1$ to generation n. During a crossover, only the first offspring (of the two generated ones) is kept.

3. Results

In this section, we present the results of the generated fusion programs on the three benchmark data sets.

The results are compared to other functions from the state of the art: (a) the *min* rule which returns the minimum score value, (b) the *mul* rule which

returns the product of all the scores, (c) the *sum* rule which returns the sum of the scores, (c) the *weight* rule which returns a weighted sum, and (d) an SVM implementation. The weighs of the weighted sum have been configured by using genetic algorithm on the training sets [50, 51] (in order to give the best results as possible). The fitness function is the value of the EER and the genetic algorithm engine must lower this value. Table 3 presents the configuration of the genetic algorithm.

Table 3: Configuration of the genetic algorithm to set the weights of the weighted sum

Parameter	**Value**
Population	5000
Generations	500
Chromosome signification	weights of the fusion functions
Chromosome values interval	$[-10; 10]$
Fitness	EER on the generated function
Selection	normalized gemetric selection (probability of 0.9)
Elitism	True

For the SVM, we have computed the best parameters (i.e., search the C and γ parameter giving the lowest error rate) using the learning database on a 5-fold cross validation scheme. We have used the *easy.py* script provided with libSVM [52] for this purpose. We have then tested the performance on the validation set. We only obtain on functional point (and not a curve) when using an SVM. That's why we have used the HTER instead of the EER.

Table 4 presents the performances, for the three databases, of each biometric systems, fusion mechanisms from the sate of the art, and our contribution.

Concerning the state of the art performances, can see that the simple fusion functions *sum* and *mul* tend to give better performances compared to the best biometric method of each database, but they are outperform by the *weight* rule. The *min* operator gives quite bad results (it does not improve the best biometric

system). The SVM method gives good results but is outperform by the $weight$ method.

Table 5 presents the gain of performance against the $weight$ operator (which gives the best results in Table 4) in term of EER and AUC.

This gain is computed as following:

$$gain = 100\frac{(EER_{weight} - EER_{gpfunc})}{EER_{weight}} \tag{2}$$

where EER_{weight} and EER_{gpfunc} are respectively the EER values of the weighted fusion and the generated score fusion function (the same procedure is used for the AUC). Better values than the weighted sum are represented in bold. The EER gives a local performance for one running point (system configured in order to obtain an FAR equal to the FRR), while the AUC gives a gives a global performance of the whole system. These two information are really interesting to use when comparing biometric systems. Figure 4 presents the ROC curves of the generated programs against the weighted sum. Performance of the initial biometric systems are not represented, because we have already seen that they are worst than the weighted sum (same remark for the other fusion functions). Logarithmic scales are used, because error rates are quite small.

We can see from Table 5 and Figure 4 that most of the time, the automatically generated functions with genetic programming give slightly better results than the weighted sum. These improvements can be local and global and vary between 16% and 59% for the EER and 0.05% and 76% for the area under the curve. When there is no improvement, the results are equal or (in one case) slightly inferior. Even if there is some difference between training (not represented in this paper) and validating sets, we cannot observe overfitting problem. The BSSR1 dataset presents the largest difference of performance between training and validation sets, but, the results are still better than the ones from the state of the art (and the same problem can be observe with the weighted sum). By the way, the fitness criterion has never been met, we did not achieve to obtain fusion functions doing no error. So, the evolution always

Table 4: Performance (HTER in %) of the initial methods (s_*^1, s_*^2, s_*^3, s_*^4, s_*^5), the state of the art fusion functions (*sum*, *min*, *mul*, *weight*) and our proposal on the three databases. Bold values represent better performance than the initial biometric systems, and * represents fusion results better than state of the art.

(a) BSSR1

Method		**HTER**
BSSR1		
Biometric systems	s_{bssr1}^1	04.30%
	s_{bssr1}^2	06.19%
	s_{bssr1}^3	08.41%
	s_{bssr1}^4	04.54%
Fusion functions	*sum*	**00.70%**
	min	05.04%
	mul	**00.70%**
	weight	**00.38%**
	SVM	**0.77%** (FAR=1.16%, FRR=0.39%)
Proposal	gpI	**0.40%**

(b) PRIVATE

Method		**HTER**
PRIVATE		
Biometric systems	$s_{private}^1$	8.92%
	$s_{private}^2$	11.53%
	$s_{private}^3$	15.69%
	$s_{private}^4$	06.21%
	$s_{private}^5$	31.43%
Fusion functions	*sum*	**02.70%**
	min	13.72%
	mul	**02.67%**
	weight	**02.26%**
	SVM	**05.47%** (FAR=10.87, FRR= 0.07%)
Proposal	gpA	**01.57%***

(c) BANCA

Method		**HTER**
BANCA		
Biometric systems	s_{banca}^1	04.38%
	s_{banca}^2	11.54%
	s_{banca}^3	08.97%
	s_{banca}^4	07.32%
Fusion functions	*sum*	**01.28%**
	min	04.38%
	mul	**01.28%**
	weight	**00.91%**
	SVM	**01.01%** (FAR= 1.71 %, FRR=0.32%)
Proposal	gpΦ	**00.75%***

Table 5: Performance gain betwain our proposal and the weighted sum (which gives the best results in the methods of the state of the art).

Database	**EER**	**AUC**
BSSR1	-5.26%	**0.05%**
PRIVATE	**34.85%**	**23.85%**
BANCA	**17.58%**	**76.74%**

ended when reaching the 50th generation.

Figure 5 represents the fitness evolution during all the generations of one genetic programming run on the BSSR1 database. A logarithmic scale has been used to give more importance to the low values and track easier the fitness evolution of the best individual of each generation. We can observe the same kind of results with the other databases. The fitness convergence appears several generations before the end of the computation. The worst program of each generation is always very bad which implies that the standard deviation of the fitness is also always quite huge. This can be explained by the high quantity of mutation probability and the low quantity of good programs kept for the next generation. When running the experiment several times, we obtain the same convergence value. We can say that we reach the maximum performance of the system.

4. Discussion

The score fusion functions generated by the proposed approach give a slightly better performance than the fusion functions used in the state of the art in multi-biometrics. We can argue that genetic programming is adapted to automatically define score fusion functions returning a score. The tradeoff of this performance gain is the need of training patterns which are not necessary for *sum*, *mul* or *min* (but this requirement is already present for the weighted sum or the use of an SVM). By the way, this is not really a problem, because we already need training patterns to configure the threshold of decision (if we do not want to do it empirically) or if we need to normalize the scores before doing the fusion.

Another problem inherent to genetic programming is the complexity of the

generated programs. It is probable that some subtrees could be pruned or simplified without loosing performance. Another trail would be to add regularization parameter to the fitness function (for example, the number of nodes or the depth of the tree). Generated programs would be more readable by an human and quicker to interpret. Figure 6 presents a simple generated tree (depending on the database, they can be more or less complex). Even if the program is quite short (comparing to the other generated functions), it includes useless code (e.g., the subtree $avg(a, a - 1/12)$ could be simplified by $a - 1/24$). Some generated trees include preprocessing steps by not using all the modalities in the terminal set.

Genetic programming generated score fusion functions give performance slightly equal or better than genetic algorithm configured weighted sum. Even if computation time is more important than for genetic algorithm, we can think that the gain is not really important between the two methods, but, to obtain these results, genetic programming needed a population ten times smaller and ten times less of generations.

5. Conclusion

We propose in this paper a new approach for multibiometrics based on the automatic generation of score fusion functions. We have seen interesting approaches in the state of the art and decided to improve them by automatically generated score fusion programs by the help of genetic programming.

Our contribution concerns the designing of multibiometric systems while using a generic approach based on genetic programming (and is inspired from the state of the art architectures). The proposed method returns a multibiometrics score to be compared with a defined threshold. The proposed multibiometric system has been heavily tested on three different multibiometric databases. We obtained great improvements compared to classical fusion functions used in the state of the art. We hope to have opened a new path in the fusion of biometric systems thanks to genetic programming.

Results could surely be improved by using different parameters in the genetic programming engine (i.e., more individuals and generations, different range of constants, different functions, ...). It could be interesting to test other performance metrics could be improved by adding quality measures of the capture, and if genetic programming could produce template fusion programs.

6. Acknowledgment

The authors would like to thank: the author of pySTEP [48], the library used during the experiment, for his helpfull help when encoutering problems with it, the authors of the various biometric databases used in this experiment, as well as the French Basse-Normandie region for its financial support.

References

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

[2] M. Hashiyada, Developement of biometric dna ink for authentication security, Tohoku J. Exp. Med. 204 (2004) 109–117.

[3] Z. Korotkaya, Biometric person authentication: Odor, Tech. rep., Department of Information Technology, Laboratory of Applied Mathematics, Lappeenranta University of Technology (2003).

[4] A. Riera, A. Soria-Frisch, M. Caparrini, C. Grau, G. Ruffini, Unobtrusive biometric system based on electroencephalogram analysis, EURASIP Journal on Advances in Signal Processing 2008 (2008) 8.

[5] R. Gaines, W. Lisowski, S. Press, N. Shapiro, Authentication by keystroke timing: some preliminary results, Tech. rep., Rand Corporation (1980).

[6] J. Fierrez, J. Ortega-Garcia, On-line signature, Springer US, 2008, pp. 189–209.

[7] A. Weiss, A. Ramapanicker, S. Pranav, S. Noble, L. Immohr, Mouse movements biometric identification: A feasibility study, in: Proceedings of Student/Faculty Research Day, CSIS, Pace University,, 2007.

[8] D. Petrovska-Delacretaz, A. El Hannani, G. Chollet, Text-independent speaker verification: State of the art and challenges, Lecture Notes In Computer Science 4391 (2007) 135.

[9] C. Nandini, C. Kumar, Comprehensive framework to gait recognition, International Journal of Biometrics 1 (1) (2008) 129–137.

[10] K. Benli, R. Duzagac, M. Eskil, Driver recognition using gaussian mixture models and decision fusion techniques, in: ISICA 2008, 2008.

[11] M. Turk, A. Pentland, Face recognition using eigenfaces, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Vol. 591, 1991.

[12] D. Maltoni, A. Jain, S. Prabhakar, Handbook of fingerprint recognition, Springer, 2009.

[13] A. Kumar, D. Zhang, Personal recognition using hand shape and texture, IEEE Transactions on Image Processing 15 (8) (2006) 2454.

[14] Z. Xu, X. Guo, X. Hu, X. Cheng, The blood vessel recognition of ocular fundus, in: Proceedings of the 4th International Conference on Machine Learning and Cybernetics (ICMLC'05), 2005, pp. 4493–4498.

[15] A. Ross, K. Nandakumar, A. Jain, Handbook of multibiometrics, Springer, 2006.

[16] M. Theofanos, B. Stanton, C. A. Wolfson, Usability & Biometrics: Ensuring Successful Biometric Systems, National Institute of Standards and Technology (NIST), 2008.

[17] ISO, Biometric performance testing and reporting, Tech. rep., ISO/IEC 1975-1:2006(E) (2006).

[18] J. Bhatnagar, A. Kumar, On estimating performance indices for biometric identification, Pattern Recognition 42 (2009) 1803 – 1815.

[19] R. Raghavendra, B. Dorizzi, A. Rao, G. Hemantha Kumar, Pso versus adaboost for feature selection in multimodal biometrics, in: IEEE 3rd International Conference on Biometrics: Theory, Applications and Systems, BTAS 2009, 2009.

[20] A. Rattani, M. Tistarelli, Robust multi-modal and multi-unit feature level fusion of face and iris biometrics, in: International Conference on biometrics (ICB2009), 2009.

[21] A. Ross, A. Jain, Multimodal biometrics: An overview, in: Proceedings of 12th European Signal Processing Conference, Citeseer, 2004, pp. 1221–1224.

[22] Y. Zuev, S. Ivanov, The voting as a way to increase the decision reliability, Journal of the Franklin Institute 336 (2) (1999) 361–378.

[23] P. Buyssens, M. Revenu, O. Lepetit, Fusion of ir and visible light modalities for face recognition, in: IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2009), 2009.

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

[25] S. Hocquet, Authentification biométrique adaptative application à la dynamique de frappe et à la signature manuscrite, Ph.D. thesis, Université de Tours (2007).

[26] L. Allano, La biométrie multimodale : stratégies de fusion de scores et mesures de dépendance appliquées aux bases de personnes virtuelles, Ph.D. thesis, Institut National des Télécommunications (2009).

[27] P. S. Teh, A. B. J. Teoh, C. Tee, T. S. Ong, A multiple layer fusion approach on keystroke dynamics, Pattern Analysis & Applications (2009) 14.

[28] 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.

[29] J. Czyz, M. Sadeghi, J. Kittler, L. Vandendorpe, Decision fusion for face authentication 7.

[30] S. Garcia-Salicetti, M. Mellakh, L. Allano, B. Dorizzi, Multimodal biometric score fusion: the mean rule vs. support vector classifiers, in: Proc. EUSIPCO, 2005.

[31] J. Koza, J. Rice, Genetic programming, Springer, 1992.

[32] A. Ross, N. Poh, Handbook of Remote Biometrics, Springer, Ch. Multibiometric Systems: Overview, Case Studies, and Open Issues.

[33] NIST, Nist biometric score set (2006).
URL `http://www.itl.nist.gov/iad/894.03/biometricscores/`

[34] N. I. of Standards, Technology, Nist biometric score set (2006).
URL `http://www.itl.nist.gov/iad/894.03/biometricscores/`

[35] E. Bailly-Bailliere, S. Bengio, F. Bimbot, M. Hamouz, J. Kittler, J. Mariéthoz, J. Matas, K. Messer, V. Popovici, F. Porée, et al., The BANCA database and evaluation protocol, Lecture Notes in Computer Science (2003) 625–638.

[36] N. Poh, Banca score database.
URL `http://info.ee.surrey.ac.uk/Personal/Norman.Poh/web/banca_multi/main.php?bodyfile=entry_page.html`

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

[38] A. Martinez, R. Benavente, The ar face database, Tech. rep., CVC Technical report (1998).

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

[40] D. Lowe, Distinctive image features from scale-invariant keypoints, International journal of computer vision 60 (2) (2004) 91–110.

[41] C. Rosenberger, L. Brun, Similarity-based matching for face authentication, in: Proceedings of the International Conference on Pattern Recognition (ICPR'2008), Tampa, Florida, USA, 2008.

[42] R. Giot, M. El-Abed, C. Rosenberger, Keystroke dynamics with low constraints svm based passphrase enrollment, in: IEEE Third International Conference on Biometrics : Theory, Applicationsand Systems (BTAS), 2009.

[43] F. Hampel, E. Ronchetti, P. Rousseeuw, W. Stahel, Robust statistics: the approach based on influence functions, John Wiley & Sons New York, 1986.

[44] A. Jain, K. Nandakumar, A. Ross, Score normalization in multimodal biometric systems, Pattern Recognition 38 (12) (2005) 2270 – 2285. URL http://www.sciencedirect.com/science/article/B6V14-4G0DDW4-1/2/d922960ee7ed8928744113dd9494d37a

[45] M. Mitchell, An introduction to genetic algorithms, The MIT press, 1998.

[46] R. Poli, W. Langdon, N. McPhee, A field guide to genetic programming, Lulu Enterprises Uk Ltd, 2008, freely available at http://www.gp-field-guide.org.uk.

[47] P. Day, A. K. Nandi, Robust text-independent speaker verification using genetic programming, IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING 15 (2007) 285–295.

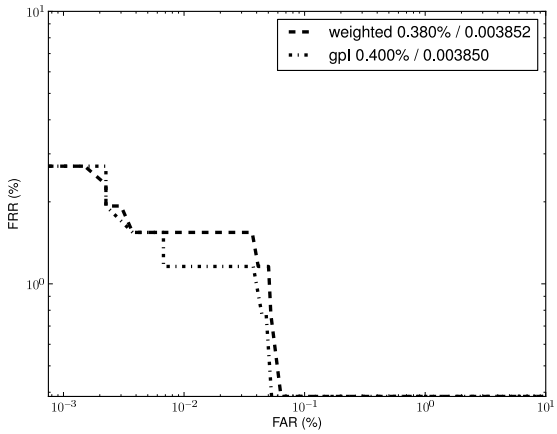
[48] M. Khoury, Python strongly typed genetic programming.
URL http://pystep.sourceforge.net

[49] T. Oliphant, Guide to NumPy, Spanish Fork, UT, Trelgol Publishing.

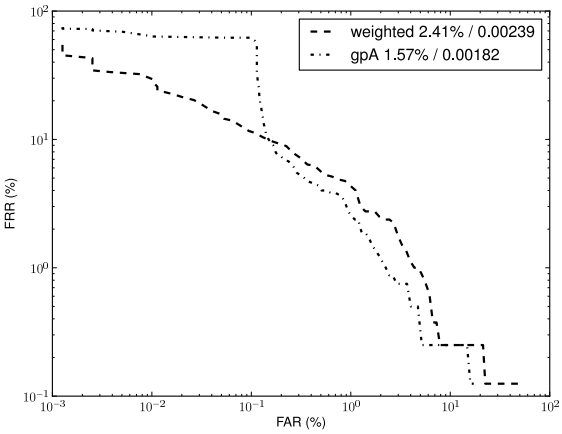
[50] R. Giot, M. El-Abed, C. Rosenberger, Fast learning for multibiometrics systems using genetic algorithms, in: The International Conference on High Performance Computing & Simulation (HPCS 2010), IEEE Computer Society, Caen, France, 2010, p. 8.

[51] R. Giot, B. Hemery, C. Rosenberger, Low cost and usable multimodal biometric system based on keystroke dynamicsand 2d face recognition, in: IAPR International Conference on Pattern Recognition (ICPR), IAPR, Istanbul, Turkey, 2010.

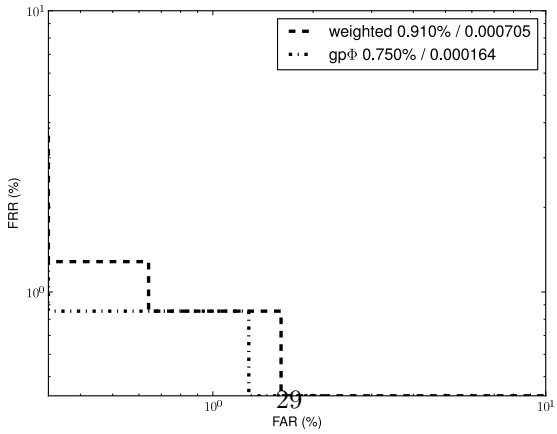
[52] C. Chang, C. Lin, LIBSVM: a library for support vector machines (2001).



(a) Validation with BSSR1



(b) Validation with PRIVATE



(c) Validation with BANCA

Figure 4: ROC curves of the fusion systems from the state of the art and with genetic programming. The EER of each fusion function is presented in the legend. Note the use of a logarithmic scale.

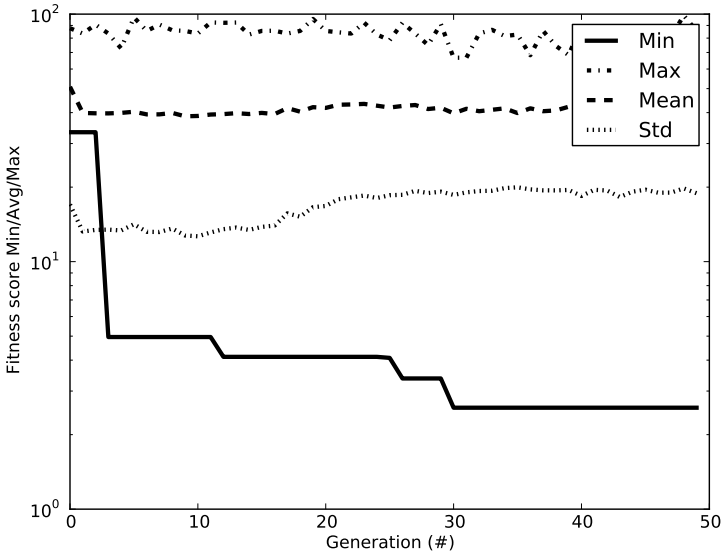


Figure 5: Fitness evolution of one run of the genetic programming evolution. The max, min, mean and std values of the fitness are represented. We want to minimize the fitness value, so lower is better.

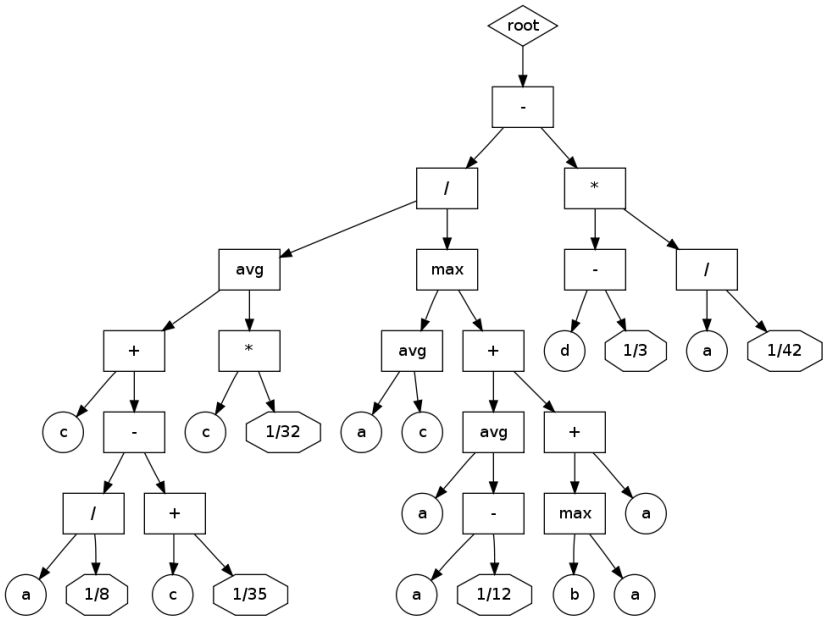


Figure 6: Sample of a "simple" generated program. We can observe the complexity of the generated fusion function.