A GA based approach for classification purposes

F. Ros, Serge Guillaume, R. Herba

A GA BASED APPROACH FOR CLASSIFICATION PURPOSES

FREDERIC ROS [1], SERGE GUILLAUME [2] AND RACHID HARBA[3]

[1]Gemalto, St Cyr en Val BP 6021 45060 ORLEANS CEDEX, [2]Cemagref, UMR Itap, France,
[3]LESI,Polytech'Orlans-LESI 12 rue de Blois 45067 0rleans CEDEX 2
e-mail: frederic.ros@gemalto.com,serge.guillaume@montpellier.cemagref.fr,rachid.harba@univ-orleans.fr
(Accepted)

ABSTRACT

This paper presents a nearest prototype classifier based on a hybrid genetic algorithm. It has been developed for general pattern recognition tasks and can be applied for identification. The approach deals simultaneously with the twofold objective of selecting relevant features and minimizing the set of instances to discard the noisy and superfluous ones. The interest of the approach is demonstrated with real life data sets.

Keywords: classification, genetic algorithms, hybrid systems.

INTRODUCTION

The purpose of an authentication system [1] is to verify the claim of the identity of an individual. It is a pattern recognition system which may operate either in verification mode ore identification mode: the first consists of a "one-to-one check". The verification is performed locally by comparing individual's features with the templates stored on the credential. The identification mode consists of presenting an individual's credentials (one-to-many check) to verify the individual exists within a known population. Identification confirms the individual is not enrolled with another identity and is not on a predetermined list of persons (e.g. blacklist). Typically this operation needs a secured database and it relies on statistical analysis and machine learning techniques [2]. The decision procedure is called a "classifier". For any biometry systems multiple templates are necessary to account for variations observed in the biometric trait and the templates (also called prototypes or instances) in the database may be updated over time. To be efficient for identification purposes, a classifier has to be tractable, selective and flexible. Compared to other well-known classifiers [3], based for example on radial basis function or back propagation neural networks, neighborhood techniques [4] remain very attractive thanks to their easy use and simple implementation. According to $1-nn$ rule, an input individual is assigned to the class of its nearest neighbor from a labeled reference set. Updating the database does not require any other computations. The main drawbacks of nearest classifiers in practice have been their computational demands and memories. In addition, they generally require low feature spaces to be efficient.The relevance of a pattern recognition system is highly dependent on measured features representing the pattern. For each database, high efficiency can be obtained only by optimizing the feature selection part. The goal of designing efficient nearest classifiers is then to maximize classification accuracy while minimizing the sizes of both prototype and feature sets. This imposes pre-processing stages to respectively edit the best prototype patterns and select the best feature vectors. Feature selection [5-6],while remaining a challenging issue, has been extensively researched. The selection can be done either considering each feature independently, the filter approach [7], or by managing a subset of the available features as in wrapper approaches [8]. The importance of the feature selection step is a crucial one as the distance function works in the feature space. The neighborhood of a given pattern is highly dependent on this distance function, i.e. the selected features. For prototypes, the objective is related to instance selection problems, editing and condensing techniques [9-10]: the goal is to select the most critical patterns in order to make the classifier faster and relevant. This objective is of prime concern when dealing with large databases as finding a pattern neighborhood requires as many distance computations as there are items in the reference data set. Despite their obvious dependence, these points are generally studied separately. In this paper, we propose to perform in a single step edition and selection via a multi objective approach, the challenge is to obtain the global optimal solution with a minimum number of experiments. Among many search algorithms, GAs [11] (genetic algorithms) are one of the best-known techniques for solving optimization problems. Their use has reported promising results in many areas and their reputation for selection problems is certain. However, the majority of the methods involved in multiobjectives suffer per nature from the excessive consuming time to design the optimal solution. Specifically, standard GA may fail and particularly when applied to real life problems involving many

features and patterns. We therefore propose a dedicated GA which is hybridized with more conventional techniques. The hybridization is structured in such a way that the classifier tractability and efficiency are both optimized. We focus here on the development of a generic classifier for small and medium databases. The next section is devoted to the methodology. We then present its performances on real pattern recognition problems to give some conclusions.

MATERIAL AND METHODS

To build an efficient nearest neighbor classifier three objectives have to be reached: achieve a high accuracy rate, minimize the set of prototypes to make the classifier tractable even with large data bases and internally, reduce the set of features used to describe the prototypes. At the opposite of many methods separating the instance and feature selection aspects despite their dependence, we propose to perform in a single step edition and selection via a multi objective approach managed by a dedicated GA. Let $Z = z_1, \ldots, z_p$ be a set of samples described by a set of features $X = x_1, \ldots, x_f$. Each item, $z_j \in R^f$, is labeled, $L = 1, \ldots, l$ being the set of available labels. Given C_{1nn} a nearest neighbor classifier, the optimization problem consists in finding the smallest subsets $S_1 \subseteq X$ and $S_2 \subseteq Z$ such that the classification accuracy of C_{1nn} over Z is maximal.

THE GENETIC ALGORITHM

GAs can be seen as powerful techniques miming natural reproduction. To solve a classification problem, a single solution via a fitness function must be presented in a single data structure. GAs will create a population of solutions based on the sample data structure proposed. In fact, they work on the basis of a set of candidate solutions. Each candidate or chromosome represents a trial solution of the problem posed and is a member of the population. For a recent review see [12]. There are few studies using evolutionary techniques to define $1-$nn classifiers with the twofold objectives of prototype and feature selection. The native ones can be found in [13-14].

Conceptual strategy: Our proposal is hybrid (See Fig.1). The genetic exploration is driven by an aggregative fitness assignment strategy. It is based on two self-controlled phases with dedicated objectives combining crowding and elitist strategies. The first one, which can be called a preliminary phase, is a pure GA. The goal is to promote diversity within the chromosome population in order to remove the unused features and to prepare the second step, called the convergence phase. This stage starts when there is a large enough number of good and diverse chromosomes in the population. The GA is then hybridized via forward and backward local procedures. The hybridization is structured in such a way that the classifier tractability and efficiency are optimized. Some neighborhood concepts related to the prototype nature are also incorporated in the local procedures. By progressively filtering useless and noisy prototypes, they contribute to select critical prototypes and discard superfluous or noisy ones. During the genetic life, elitism and pressure preservation are reinforced by two mechanisms, say, a breaking process and an evolutionary memory. Instead of trying in vain to maintain an effective search and a good selection pressure in the current population, this mechanism considers the current best chromosome for a secondary population (called the archive population) while re-seeding the current population to explore new directions. The archive population is updated at each generation from the chromosome solutions found out through the different explorations in such a way that both diversification and elitism are promoted. The combination of implemented mechanisms and the particular hybridization approach constitute the main parts of our contribution. It should be noted that the first phase is mandatory for starting the design of a new classifier but not necessary when new individuals are added to the native database. In this case, these individuals are included in preliminary "competent" chromosomes to form the initial population of the second stage.

The chromosome: As the optimization procedure deals with two distinct spaces, the feature space and the pattern space, both are managed by the GA. A chromosome represents the whole solution. It is encoded as a string of bits, whose length is f+p, f being the number of available features and p the number of patterns in the training set. In a chromosome a 1 for the ith feature or pattern stands for its selection, while a 0 means it is not taken into account. As the number of features is likely to be smaller than the number of patterns, in order to speed up the procedure and to improve the exploration power of the algorithm, the two spaces are managed independently at each iteration by the genetic operators such as crossover and mutation. This means the whole chromosome is the union of two distinct subchromosomes, the first one to encode the feature space and the second one the pattern space. In each subchromosome a classical one-point crossover is applied. We superimpose some general restrictions for a chromosome to represent a valid solution. Some others can be included for each particular problem.

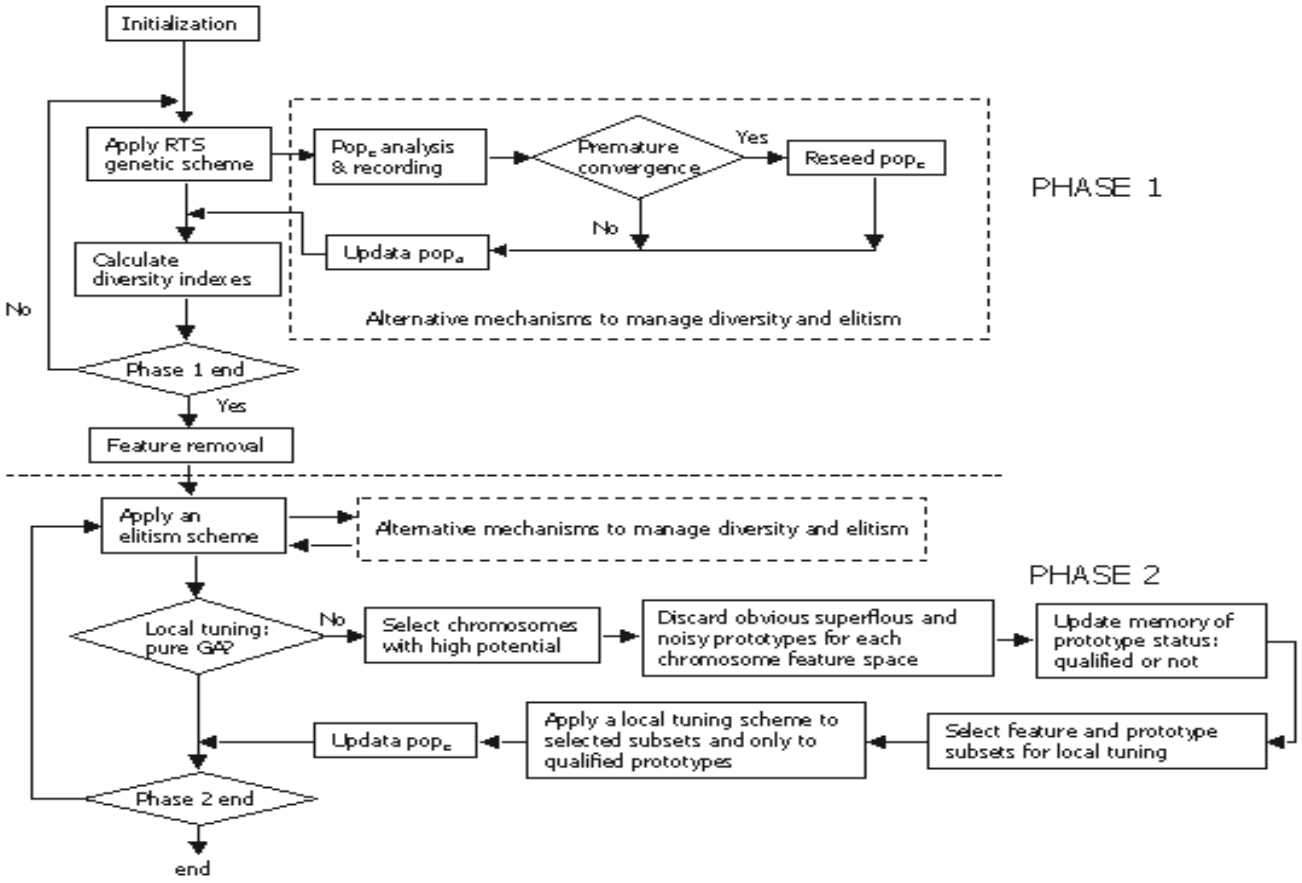

Fig. 1. *Hybrid algorithm: basic diagram*

Fitness function: The choice of the fitness function is of prime importance in a GA design. A aggregative fitness assignment strategy has been selected:

$$F = w_c C_{1nn}(Z) + w_f \lambda_f + w_p \lambda_p \qquad (1)$$

with $w_c + w_f + w_p = 1$. The weight values stand for the importance of the corresponding objectives. The parameters λ_f and λ_p, have to be maximal for a small number of selected features or patterns.

$$\lambda_f = \begin{cases} 1 - \frac{|S_1|}{f_{max}} & if\ |S_1| \leq f_{max} \\ 0 & else \end{cases} \qquad (2)$$

$$\lambda_p = \begin{cases} 1 - \frac{|S_2|}{p_{max}} & if\ |S_2| \leq p_{max} \\ 0 & else \end{cases} \qquad (3)$$

The parameter f_{max} for the feature space and p_{max} for the pattern space are set to limit the range of variation of l.

SPECIAL MECHANISMS

Preserving both elitism and diversity constitutes the main challenge for a GA. Most methods [14-15] such as Determinist Crowding (DC), Restricted Tournament Selection (RTS) and others are continuously looking for a balance between elitism and diversity in the current population. We have implemented two mechanisms:

An evolutionary memory: Two distinct populations with different evolution rules and no direct interaction are used. The first one is called the current population, pop_c, its evolution is managed using classical genetic schemes (elitism, DC, RTS, ...). The second one is called the archive population, pop_a, it acts as an evolutionary memory. It is a repository of good chromosome solutions found during the evolution. The chromosomes of pop_c are included in pop_a only if they provide either more elitism or more diversity. The decision to include a given chromosome in popa is based on two criteria, the first one is the fitness score. This is the elitist side of the process. If the candidate score is slightly better than others, the candidate replaces the chromosome with the most comparable structure. This is the diversity side of the process.

Breaking process: The current population needs to be reseeded when there are a lot of similar chromosomes within the population. The similarity between the ith and jth chromosomes is:

$$s(i,j) = \begin{cases} 1 & if\ d_h^f(i,j) < n_f\ and\ d_h^p(i,j) < n_p \\ 0 & else \end{cases} \quad (4)$$

where $d_h^f(i,j)$ (resp. $d_h^p(i,j)$) stands for the hamming distance in the feature (resp. pattern) space, and n_f (resp. n_p) is a predefined threshold. The proportion of chromosomes similar to the ith one is given by:

$$P_s(i) = \frac{1}{s-1} \sum_{j=1, j\neq i}^{s} s(i,j) \quad (5)$$

where s is the population size. This mechanism produces major changes in the current population by including chromosomes from the archive population or applying a high mutation rate to refresh the chromosome. The breaking mechanism is active when there are a lot of similar chromosomes within the population. The $P_s(i)$ are thresholded to compute the diversity index:

$$DI = \frac{1}{s}\sum_{i=1}^{s} S(i)\ \ where\ \ S(i) = \begin{cases} 1 & if\ P_s(i) > th_{min} \\ 0 & else \end{cases} \quad (6)$$

When the diversity index, DI, is too low, some of the chromosomes which have a lot of similar ones in the population, some of the i ones for which $S(i) = 1$, are either replaced by ones randomly chosen in the archive population or re-generated with a high mutation probability.

OPTIMIZATION

Convergence phase: in this phase, the GA is intelligently hybridized via ascending and descending local phases. The ascending one aims at aggregating new elements, features or prototypes, in a given chromosome while the goal of the descending phase is, on the contrary, to remove features or prototypes from the chromosome description. Both procedures are random free. They are based on the population yielded by the GA. The hybridization is applied in such a way that the contribution of local optimizations is relevant for optimization without monopolizing too many computing resources, making the method practical for medium size data. The local procedures are carried out periodically in sequences and only a fraction of the chromosomes randomly chosen and only a variable subset of chromosome components are considered. They are applied only with competent chromosomes to avoid vain computational processing. It is a good compromise between the complete use of local search, which is not practicable, and complete removal, which leads to worse solutions.

Local optimization: Let us first consider the ascending step. It can be applied to the feature or the prototype space. Let S' be the set of chromosomes whose fitness score is higher than a given threshold, and $S'_1 \subseteq X$ (resp. $S'_2 \subseteq Z$) be the set of features (resp. prototypes) included in at least one chromosome (from S') description. The ascending procedure consists, for each chromosome in S', in aggregating each of the features in S'1 (resp. each of the prototypes in S'2) to the chromosome and selecting the ones that improve the classification results. The process is repeated until no improvement is possible or a maximal number of ascendant iterations is reached. It should be mentioned that the number of features and prototypes to be tested is reasonably small as some features have been discarded by the first phase of the GA, and among the others, only those which are currently part of one of the best chromosomes are used. This remark highlights the complementary roles played by the GA and the local approach. However, depending on the evolution stage, the cardinalities of S'1 and S'2 may be important. In this case, in order to control the ascendant procedure computational cost, the number of features or prototypes tested by the procedure is limited. The selected ones are randomly chosen in S'1 or S'2. The descending phase is only applied to the S' set. For each chromosome each of the selected features (resp. prototypes) is removed if its removal does not affect the classification results while improving the fitness function.

Prototype filtering: The prototype selection is not only based on classification results, it also takes into account the prototype status within the training set in order to avoid selecting either noisy or superfluous prototypes and favor the selection of critical ones. A simple and unambiguous definition has been considered. A prototype is said to be noisy if none of its k nearest neighbors is of its class. That means this prototype is not able to correctly classify any of its neighbors. The value of k, $k \geq 1$, is set according to the class cardinality, the higher k the lower the number of prototypes likely to be removed. On the other hand, a prototype is said to be superfluous if all of its neighbors are of its class. That means its neighbors remain well classified after its removal. The amount of filtering depends only on the number of neighbors, k, no additional heuristic is needed. These concepts, noisy or superfluous prototypes, are highly dependent on the feature space and the distance in use. Thus, their

Table 1. *Data base characteristics*

	Iris	Breast	Gaussian8D	Satimage	Texture	Chemo1	Chemo2
# items	150	699	1000	1028	989	566	1294
# features	4	9	8	36	40	166	167
# classes	3	2	2	6	11	8	8

implementation requires a specific management via an evolutive memory: a list of prototypes to be discarded is attached to a given feature space. The more the convergence phase progresses, the more the memory provides information about the prototype nature and speed up the local procedures. The identification of noisy or superfluous prototypes is carried out at the beginning of the descending procedure. The prototypes part of the list are no longer available, neither for the ascending procedure nor GA selection or chromosome generation. Note that GA may select superfluous prototypes as they improve classification results. In this case the solution found can be considered as good but remains a suboptimal one.

RESULTS AND DISCUSSION

The method is not dedicated to a specific application domain. It has been initially developed for pattern recognition applications involving few individuals (classes) with many templates. We therefore have then devoted the evaluation to this context by selecting seven known data sets of variable difficulties. Five have been largely used to test many machine learning algorithms [8] and the two others are from the chemometric area. For some of them, we have sampled the individuals to respect a balance in the sizes. The database characteristics are summarized in Table 1.

From each data set, 10 training and test samples are randomly generated. The training set is made up of about 80% of the data set, the remaining 20% being the test set. For each of the ten samples, the training data are centered and normalized, the computed coefficients are applied to the corresponding test set. For each database, we compare our hybrid GA to two very popular genetic approaches: MHA (Multi Hill Climbing) and EA (Elitism strategy) algorithms [13-14]. Both are not hybrid. These methods were compared to no genetic approaches where selection and edition were applied in different steps. They have proved to be better than all the other combinations of tested approaches. We then focus on comparing our hybrid GA to only genetic approaches. Then for each of the 3 algorithms, a classifier is designed using the training data and its performance is assessed over the test data. The same very common genetic parameters have been chosen whatever the database, and of course the same fitness function. The selected weights and other parameters have been the followings: w_c=0.4, w_p=w_f=0.3, n_f=1, n_p=0.1*p, th_{min}=0.75*s. The overall results (Table 2)correspond to the best overall score obtained after 20 runs of 300 generations for each algorithm and a population of 100 chromosomes. Each element respectively depicts the number of features, instance patterns, and classification test scores. These results highlight that the HGA gives better results than any of the other algorithms. They vary a lot regarding database difficulty. For very simple sets such as iris or breast, there is no real difference. The native feature space is already competent for classification purposes without any selection. The databases Gaussian 8D, Satimage and Texture present more features and patterns. HGA leads every time to more tractable and efficient classifiers. Chemometric databases are reputed difficult to manage as they contain more overlapping. HGA gives various competent chromosomes and its superiority is globally higher than the ones of MGA and EA. It is clear that only HGA is able to perform the double selection correctly. It should be noted that the classifier performances obtained for Chemo 2 can be improved by increasing a little w_c. With w_c=0.6 providing more importance to the efficiency, the classification performances are between 0.8 and 0.85 with a cardinality for S_1 and S_2 less than 5. Concerning the other approaches, MGA has more difficulties than the others to manage the three objectives. It can be easily explained: This random recursive search approach does not include intelligent mechanism. EA is a selective algorithm which can come across a classical premature convergence issue due to the difficulty to define the right controlling parameters (which remains a challenge for pure GAs). This consolidates our idea of introducing different helping mechanisms (evolutionary memory, breaking process, hybridization) to compensate some weaknesses of pure GAs.

CONCLUSION

In this paper, we have proposed a nearest classifier based on a hybrid algorithm aiming at maintaining

Table 2. *Method Evaluation*

	Iris	Breast	Gaussian8D	Satimage	Texture	Chemo1	Chemo2
EA	1-3-0.96	1-31-0.91	2-71-0.71	3-80-0.8	4-36-0.82	16-48-0.59	36-100-0.81
MGA	1-15-0.94	2-110-0.91	2-149-0.65	6-149-0.81	6-143-0.75	21-93-0.55	32-90-0.71
HGA	1-3-0.97	1-7-0.94	2-2-0.78	2-2-0.81	3-3-0.88	2-2-0.65	3-3-0.73

both qualities of a genetic population, namely diversity and elitism. We have showed with real life databases that it produces better results than two very popular GA which had already proved their superiority against classic pattern recognition schemes. Some specificities and optimizations will be incorporated to make it suitable for application fields such as biometric face recognition where it is necessary for each individual to enroll many templates to obtain efficient classifiers. Generalization to very large databases can be done by adding data mining concepts. Similarly, it is possible to perform the classifier training using divide and conquer approaches. They consist in dividing a large problem into smaller sub problems to merge the sub-solutions into the final one.

REFERENCES

Jain A.K., Ross A. and Prabhakar S. 2004 *IEEE Trans. Cir. Syst.***(14)1** 4-20

Brighton H., Mellish C. 2002 *Journal of Data Mining and Knowledge Disc.* **6** 153-172

Ros F., M.Pintore, J.R. Chretien 2007 *Journal of chem. and intelligent laboratory systems* **87(2)** 231-240

Dasarathy B. V., Sanchez J. S. and Townsend S. 2003 *Pattern Anal. and Appl.* **3** 19

Zongker D. and Jain A.K. 2004 *IEEE Trans. On Pattern Analysis and Machine Intelligence* **26 (9)** 1105-1113

A.Blum and P.Langley 1997 Machine Learning Artif. Intell **97(1-2)** 245-271

Piramuthu S. 2004 *European Journal of Operational Research* **156** 483-494

Kohavi R and John G. 1997 *Artificial Intelligence* 273-324.

Brighton H. and Mellish C 2002 Data Mining and Knowledge Discovery **6** 153-172

Wilson D.R and Martinez T.R 2000 Machine Learning **38(3)** 257-286

Goldberg D.E. 1989 *Genetic Algorithms in Search,Opt. and Machine Learning)*(Addison-Wesley, Boston) 35.

Goldberg D. and Kumura S. 2007 *Genetic Algorithms: The design of Innovation*(Springer-Verlag, New York)

Shalak D.B 1994 In Proc. of the Eleventh Inter. Conf. on Mach. Learning (Morgan Kaufman) 293

Kuncheva L.I, Jain L.C. 1999 *Journal of Pattern Recognition Letters* **20 (11)** 1149-1156

Blake C., Keogh E., Merz C.J. 1998 *[http://www.ics.uci.edi/ mlearn/MLRepository.html]*