Dominant Genetic Algorithm for Feature Selection with Associative Models

José A. Estrada-Pavía, Mario Aldape-Pérez, and Oscar Camacho-Nieto

Centro de Innovación y Desarrollo Tecnológico en Cómputo (CIDETEC)
Instituto Politécnico Nacional (IPN)
Ciudad de México, México
jestradap@ipn.mx; maldape@ipn.mx; oscarc@cic.ipn.mx;
http://www.aldape.mx

Abstract. In machine learning and statistics, feature selection, is the process of selecting a subset of relevant features that helps to decrease the dimension of a set of patterns and to improve classification performance. Feature Selection aims to obtain the least subset of features that represent the problem In this paper a novel algorithm is presented. It is called Dominant Genetic Algorithm (DGA). This genetic algorithm is a wrapper style feature selection algorithm and the search strategy is guided by an hybrid associative classifier. The classifier evaluates different solutions and gives them a value based on classification accuracy. Obtained results suggest that the algorithm can be used to improve classification performance. Experimental phase was performed using five data sets, widely used in machine learning and statistics.

Keywords: Associative models, CHAT, DGA, Genetic Algorithms, Pattern Recognition.

1 Introduction

Every day more data is generated, but all that data has no meaning until a process transform it into information. One way to obtain information from data is by classifying them. When the data is classified, it is desirable to have as less characteristics as possible, this means that is desirable to perfom a feature selection process before classifying.

The best way to take on feature selection is with an exhaustive search of the best subset of characteristics, this can be done only in datasets where there are not many characteristics. In those datasets with huge amount of features, others methods, instead of exhaustive searching must be looked for [1].

Feature selection algorithms may roughly be divided into three types: The first type is the set of algorithms that are build into adaptive systems for data analysis, called predictors. The second type are wrapper around predictors,

providing them a subset of features and receiving their feedback, those algorithms are known as wrapper algorithms. The third type are independent algorithms from predictors, these algorithms filter the data that have little chance to be useful in classification, these filter methods are based on performance evaluation metric, calculated directly from the data [2].

Feature selection can be analyzed as a search process, where the search must be guided by the classification performance and also by the number of selected characteristics. Search processes are optimization problems that can be solved using genetic algorithms. The use of a genetic algorithm to solve the problem of feature selection was introduced by Siedlecki and Sklansky[3]. Also Ludmila Kuncheva used a genetic algorithm in feature selection for parallel classifiers [4,5].

In this paper, a genetic algorithm wrapper-based method for feature selection is proposed, the DGA. This method employs an associative classifier as the method for evaluating feature subsets. The use of a wrapper-style feature selection method with an associative classifier was first proposed in [6]. This proposal executes an exhaustive search using parallel processing to improve the classification performance. However, that approach is still too time-consuming. This work aims to analyze a suitable algorithm to improve the feature selection in big data sets. The rest of the paper is organized as follows. Section 2 provides a brief overview of genetic algorithms. Section 4 explains the associative classifier employed in the proposed model. The proposed model is described in Section 5 and the experimental results are presented in Section 6. To conclude, Section 7 presents a discussion on the obtained results.

2 Genetic Algorithms

Genetic Algorithms (GA) are search procedures based on natural selection and natural genetics. The first GA was developed by John H. Holland in 1960 to allow computers to evolve solutions to difficult search and combinatorial problems, such as function optimization and machine learning [7]. Also the genetic algorithm can be defined as a highly parallel mathematical algorithm that transforms a set (population) of individual mathematical objects, each with an associated fitness value, into a new population using operations patterned after the Darwinian principle of reproduction and survival of the fittest and after naturally occurring genetic operations (notably sexual recombination) [8].

Recently GA have been applied in different areas like in transit network design problem [9], projects of public illumination [10] and in text classification [11], among others.

It is said that most of GA methods have at least four elements in common: Population of chromosomes, selection according to fitness, crossover to produce new offspring, and random mutation of new offspring. The chromosomes are commonly represented as a binary array with n alleles that can get the values

0 or 1. Each chromosome can be thought of as a point in the search space of candidate solutions. The GA processes populations of chromosomes, successively replacing one population with another. The GA requires a fitness function that assigns a score to each chromosome in the current population. The fitness of a chromosome depends on how well that chromosome solves the problem at hand.

The simplest form of genetic algorithm involves three operators: selection, crossover and mutation, each operator is explained below:

Selection: The function of this operation is to select elitist individuals as parents in current population, which can generate offspring. Fitness values are used as criteria to judge whether individuals are elitist [12]**.** The purpose of selection is to emphasize the fitter individuals in the population hoping that their offspring will in turn have even higher fitness [13].

Crossover: This operator randomly chooses a locus and exchanges the subsequences before and after that locus between two chromosomes to create two offspring. For example the chromosomes 11110000 and 10100011 could be crossed over by the midpoint of each, to produce the two offspring 11110011 and 10100000. The objective of this operator is to recombine building blocks (schemas) on the different solutions.

Mutation: This operator randomly flips some of the bits in a chromosome. For example, the string 00000100 might be mutated in its second position to yield 01000100. Mutation can occur at each bit position in a string, with some probability, usually very small (e.g., 0.001). Mutation ensures the population against permanent fixation at any particular local result.

According to De Jong [14] a standard GA is:

- Randomly generate a population of m parents.
- Repeat:
  - Compute and save the fitness u(i) for each individual i in the current parent population.
  - Define selection probabilities p(i) for each parent i so that p(i) is proportional to u(i).
  - Generate m offspring by probabilistically selecting parents to produce offspring.
  - Select only the offspring to survive.
- End Repeat

3 Associative Memories

An associative memory $\mathbf{M}$ is a system that relates input patterns and output patterns as follows:

$$\mathbf{x} \rightarrow \boxed{\mathbf{M}} \rightarrow \mathbf{y}$$

with $\mathbf{x}$ and $\mathbf{y}$ the input and output pattern vectors, respectively. Each input vector forms an association with its corresponding output vector. For each γ

integer and positive, the corresponding association will be denoted as: $(\mathbf{x}^\gamma, \mathbf{y}^\gamma)$. An associative memory $\mathbf{M}$ is represented by a matrix whose ij-th component is m_{ij}. An associative memory $\mathbf{M}$ is generated from an *a priori* finite set of known associations, called the fundamental set of associations. If μ is an index, the fundamental set is represented as: $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$ with p as the cardinality of the set. The patterns that form the fundamental set are called fundamental patterns. If it holds that $\mathbf{x}^\mu = \mathbf{y}^\mu \ \forall \mu \in \{1, 2, ..., p\}$, $\mathbf{M}$ is autoassociative, otherwise it is heteroassociative; in this case, it is possible to establish that $\exists \mu \in \{1, 2, ..., p\}$ for which $\mathbf{x}^\mu \neq \mathbf{y}^\mu$. If we consider the fundamental set of patterns $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$ where n and m are the dimensions of the input patterns and output patterns, respectively, it is said that $\mathbf{x}^\mu \in A^n$, $A = \{0, 1\}$ and $\mathbf{y}^\mu \in A^m$. Then the j-th component of an input pattern $\mathbf{x}^\mu$ is $x_j^\mu \in A$. Analogously, the i-th component of an output pattern $\mathbf{y}^\mu$ is represented as $y_i^\mu \in A$. Therefore, the fundamental input and output patterns are represented as follows:

$$\mathbf{x}^\mu = \begin{pmatrix} x_1^\mu \\ x_2^\mu \\ \vdots \\ x_n^\mu \end{pmatrix} \in A^n \qquad \mathbf{y}^\mu = \begin{pmatrix} y_1^\mu \\ y_2^\mu \\ \vdots \\ y_m^\mu \end{pmatrix} \in A^m$$

A distorted version of a pattern $\mathbf{x}^\gamma$ to be recalled will be denoted as $\widetilde{\mathbf{x}}^\gamma$. An unknown input pattern to be recalled will be denoted as $\mathbf{x}^\omega$. If when an unknown input pattern $\mathbf{x}^\omega$ is fed to an associative memory $\mathbf{M}$, happens that the output corresponds exactly to the associated pattern $\mathbf{y}^\omega$, it is said that recalling is correct [15].

4 CHAT Associative Memory

CHAT Associative Memory is based on the combination of two associative memories, the Lernmatrix from Steinbuch [16,17] and Linear Associator from Kohonen [18]. The algorithm is as follows:

1. Lets define a fundamental set of input patterns of dimension n with real values on their components, these patterns are organized in m different classes.
2. For each input pattern in the class k define a vector made of zeros except the $k - th$ coordinate, where the value is one.
3. Obtain the medium vector from the fundamental set of patterns according to the following expression:

$$\overline{\mathbf{x}} = \frac{1}{p} \sum_{\mu=1}^{p} \mathbf{x}^\mu \tag{1}$$

With $\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, ..., \mathbf{x}^p$ as the set of input patterns
With $\bar{\mathbf{x}}$ as the medium vector for all input patterns
With p as the total number of patterns to be used

4. Take the components from the medium vector as the center of a new set of coordinates.
5. Move all the patterns from the fundamental set according to the following expression:

$$\mathbf{x}^{\mu'} = [\mathbf{x}^{\mu} - \overline{\mathbf{x}}] \ \ \forall \mu \in \{1, 2, ..., p\} \tag{2}$$

With $\mathbf{x}^{\mu'}$ as the displaced pattern
With $\mathbf{x}^{\mu}$ as the original pattern
With $\bar{\mathbf{x}}$ as the medium vector for all input patterns

6. Obtain an associative memory $\mathbf{M}$ by performing the following steps:
   - Given the fundamental set of associations $\{(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}) \mid \mu = 1, 2, ..., p\}$, obtain the displaced fundamental set of associations $\{(\mathbf{x}^{\mu'}, \mathbf{y}^{\mu'}) \mid \mu = 1, 2, ..., p\}$ using expression (1) and expression (2).
   - Consider each one of the p associations $(\mathbf{x}^{\mu'}, \mathbf{y}^{\mu'})$, so an $m \times n$ matrix is obtained according to the following expression:

$$\mathbf{y}^{\mu'} \cdot \left(\mathbf{x}^{\mu'}\right)^t = \begin{pmatrix} y_1^{\mu'} \\ y_2^{\mu'} \\ \vdots \\ y_m^{\mu'} \end{pmatrix} \cdot \left(x_1^{\mu'}, x_2^{\mu'}, ..., x_n^{\mu'}\right) \tag{3}$$

$$\mathbf{y}^{\mu'} \cdot \left(\mathbf{x}^{\mu'}\right)^t = \begin{pmatrix} y_1^{\mu'} x_1^{\mu'} & \cdots & y_1^{\mu'} x_j^{\mu'} & \cdots & y_1^{\mu'} x_n^{\mu'} \\ \vdots & & \vdots & & \vdots \\ y_i^{\mu'} x_1^{\mu'} & \cdots & y_i^{\mu'} x_j^{\mu'} & \cdots & y_i^{\mu'} x_n^{\mu'} \\ \vdots & & \vdots & & \vdots \\ y_m^{\mu'} x_1^{\mu'} & \cdots & y_m^{\mu'} x_j^{\mu'} & \cdots & y_m^{\mu'} x_n^{\mu'} \end{pmatrix}$$

   - Obtain an associative memory $\mathbf{M}$ by adding all the p matrices according to the following expression:

$$\mathbf{M} = \sum_{\mu=1}^{p} \mathbf{y}^{\mu'} \cdot \left(\mathbf{x}^{\mu'}\right)^t = [m_{ij}]_{m \times n} \tag{4}$$

in this way the ij-th component of an associative memory $\mathbf{M}$ is expressed as follows:

$$m_{ij} = \sum_{\mu=1}^{p} y_i^{\mu'} x_j^{\mu'} \tag{5}$$

7. Apply the recalling phase as follows:

The recalling phase pretend to find the class label that belongs to an unknown input vector $\mathbf{x}^{\omega} \in A^n$. Find the class means to obtain the components from the vector $\mathbf{y}^{\omega} \in A^p$ that belongs to the pattern $\mathbf{x}^{\omega}$. The recalled output pattern is obtained according to the following expression:

$$y_i^{\omega} = \begin{cases} 1 \text{ if } \sum_{j=1}^{n} m_{ij} \cdot x_j^{\omega} = \theta \\ 0 \quad \text{otherwise} \end{cases} \tag{6}$$

5 Proposed Method

This paper presents a wrapper-style feature selection algorithm based on a Genetic Algorithm and an associative classifier. The objective of this algorithm is to achieve a solution near to the optimal, exploring as less solution space as possible. The proposed feature selection method is described in Fig. 1.

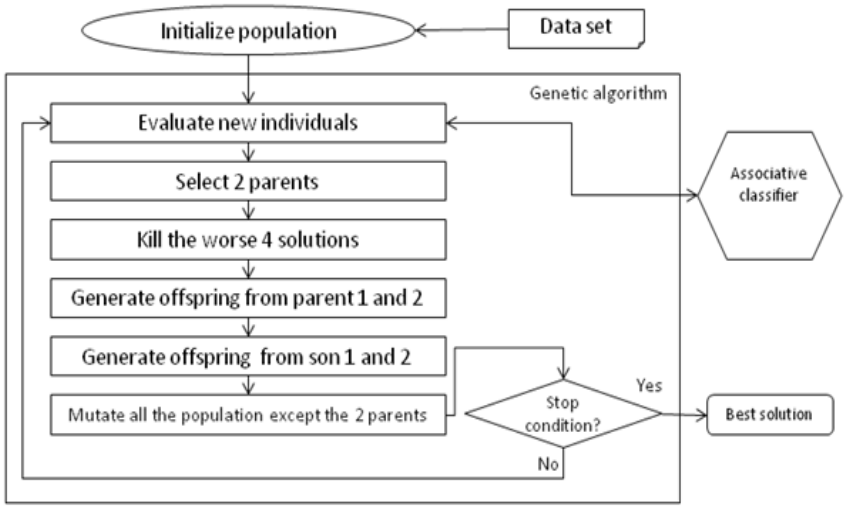

Fig. 1: Proposed method

As can be seen on the Fig. 1 the proposed algorithm is different from conventional GA, first in DGA an initial population of 6 is chosen randomly, this

Table 1: Characteristics of datasets used in the experimental phase.

Data set	No. of features	Solution space
Breast Cancer	9	512
Heart	13	8,192
Hepatitis	19	524,288
Lung Cancer	56	1.44E17
Arrhythmia	279	9.71E83

little population helps to decrease the number of the solutions explored. All the individuals are binary coded, where 1 represents the use of a specific characteristic and 0 represents a characteristic that is not selected. Instead of selecting parents randomly, the best two solutions are always selected, using a simple linear ranking selection. In this method, chromosomes of a population are sorted according to their fitness values [19].

The Fitness function that is proposed has two objectives. The first is to increase the classification performance and the second is to reduce the number of selected features. Even though both parameters are crucial to increase the performance of a machine learning model, classification performance is more important than the number of selected features. Considering the previous, a Fitness function is proposed.

$$Fitness(x, y, z) = y * .97 + (100 - ((\frac{100}{z}) * x)) * .03 \quad (7)$$

Where:

x: Is the selected characteristics

y: Is the classification performance

z: Is the total number of features

The values 0.97 and 0.3 were assigned in order to give a greater weight to higher classification rates and less weight to dimensionality reduction.

The crossover operator is a one-point crossover, the spot to make the crossover is exactly or near in the middle of the chromosome.

The mutation operator, mutates all the population excluding the selected parents, the mutation probability is 0.18 for each allele in the solution.

6 Experimental Results

Along the experimental phase, five datasets taken from the UCI Machine Learning Repository [20] were used. Specifically, the Breast Cancer, Heart disease, Hepatitis diagnosis, Lung Cancer and Arrhythmia data sets were chosen. The characteristics of these datasets are shown in Table 1.

Four algorithms were proposed to make a comparison with the DGA:

Table 2: Mutation rates.

Algorithm	Mutation rate
SGA1	4%
SGA2	26%
SGA3	6%
SGA4	27%

- Simple Genetic Algorithm with Binary Tournament selection method and Singlepoint crossover method called SGA1.
- Simple Genetic Algorithm with Stochastic Remainder Sampling selection method and Singlepoint crossover method called SGA2.
- Simple Genetic Algorithm with Binary Tournament selection method and Doublepoint crossover method called SGA3.
- Simple Genetic Algorithm with Stochastic Remainder Sampling selection method and Doublepoint crossover method called SGA4.

Different mutation rates were used in each algorithm in order to improve performace of each one. Mutation rate of each algorithm is shown in Table 2.

Each algorithm was run 20 times. The search process stopped when the number of evaluated indivuduals got 5% of the solution space in Breast Cancer, Heart and Hepatitis datasets. In Lung Cancer and Arrhythmia datasets the search process stopped when the number of evaluated indivuduals was 100.

Table 3 shows classification performance and dimensionality reduction in each dataset. Near Optimum indicates how far was the solution obtained by genetic algorithms compared to the optimal solution, which is obtained by brute force. After exhaustive exploration of the solution space, the selected subset of features that maximizes classification accuracy is selected. In case that two or more subset of features achieve the same classification accuracy, the subset of features with the smallest number of them is selected.

7 Conclusions

This study presents a genetic algorithm-based approach called DGA. The classical GA operators have been taken and mixed, in a new way to solve a known problem, the feature selection. The results showed that this approach to feature selection obtains an approximately optimal solution in less time than an exhaustive search would need; especially so on data sets where the amount of features starts to prohibit full exploration of the solution space. This suggests that the DGA is a useful tool for feature selection, especially for larger datasets.

The results of the conducted experiments show that the DGA coupled with an associative classifier exhibits a good performance using a reduced solution space.

Table 3: Classification accuracy using 10 fold cross-validation.

		Arrhythmia	Breast Cancer	Heart	Hepatitis	Lung Cancer
DGA	CA	**57.95%**	**96.90%**	**80.15%**	**84.10%**	**81.30%**
	DR	53.53%	41.11%	72.31%	**73.95%**	**71.52%**
	NO		**100%**	**97.39%**	**99.46%**	
SGA1	CA	55.60%	96.45%	73.15%	78.15%	79.05%
	DR	**54.16%**	**46.11%**	**75.38%**	60.53%	64.38%
	NO		85.41%	75.52%	94.38%	
SGA2	CA	51.50%	96.20%	70.40%	68.30%	71.60%
	DR	49.68%	42.78%	51.15%	54.21%	51.07%
	NO		85.41%	66.14%	59.62%	
SGA3	CA	54.45%	96.75%	75.15%	77.25%	78.90%
	DR	52.03%	40%	67.31%	62.89%	59.91%
	NO		85.41%	82.29%	91.71%	
SGA4	CA	51.65%	96.20%	70.90%	68.05%	72.85%
	DR	49.87%	41.11%	53.46%	53.68%	50.80%
	NO		85.41%	69.27%	59.62%	

CA: Classification accuracy, DR: Dimensionality reduction, NO: Near Optimum

The experimental results suggest that it is a feasible approach for datasets consisting of more than a dozen features, obtaining good classification performance values with an important feature amount reduction evaluating a little number of solutions.

It is clear from these results that the DGA is capable of approximating the optimal feature subset within less of 5% in average sized datasets. Regarding to dimensionality reduction, the results show important cut down on the number of features selected. For bigger datasets, a reduction of half or more of the features can be observed.

The key result observed in these experiments is the reduction of the searching space. It can be seen from the result tables that even the most inefficient experiments (the ones on the Breast Cancer dataset) required less evaluations than an exhaustive search to find the optimal solution or one extremely similar to it in performance.

Further work can be done trying to improve the operators used to get an even better solution. Additionally, different genetic operators can be used to find the combination of these which yields better results. Other kinds of evolutionary algorithms should be tested in the future in order to determine if they can produce similar or better results than the DGA; similarly, future research should also focus on coupling the DGA with other classifiers and prove its efficiency.

Also a future research should be done in order to know, the percentage needed to be evaluated in the solution space, to get an almost optimal solution, in big data sets.

References

1. Aldape-Pérez, M., Yánez-Márquez, C., Camacho-Nieto, O., Argüelles-Cruz, A.J.: An associative memory approach to medical decision support systems. Computer Methods and Programs in Biomedicine **106** (May 2012) 287–307
2. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3** (2003) 1157–1182
3. Siedlecki, W.W., Sklansky, J.: A note on genetic algorithms for large-scale feature selection. Pattern Recognition Letters **10**(5) (1989) 335–347
4. Kuncheva, L.I.: Fitness functions in editing k-nn reference set by genetic algorithms. Pattern Recognition **30**(6) (1997) 1041–1049
5. Kuncheva, L., Jain, L.C.: Designing classifier fusion systems by genetic algorithms. IEEE Trans. Evolutionary Computation **4**(4) (2000) 327–336
6. Aldape-Pérez, M., Yáñez-Márquez, C., Camacho-Nieto, O., Ferreira-Santiago, Á.: Feature selection using associative memory paradigm and parallel computing. Computación y Sistemas **17**(1) (2013) 41–52
7. Marinakis, Y., Dounias, G., Jantzen, J.: Pap smear diagnosis using a hybrid intelligent scheme focusing on genetic algorithm based feature selection and nearest neighbor classification. Computers in Biology and Medicine **39**(1) (2009) 69 – 78
8. Koza, J.R., Banzhaf, W., Chellapilla, K., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M.H., Goldberg, D.E., Iba, H., Riolo, R.L.: Genetic programming 1998: Proceedings of the third annual conference. IEEE Trans. Evolutionary Computation **3**(2) (1999) 159–161
9. Nayeem, M.A., Rahman, M.K., Rahman, M.S.: Transit network design by genetic algorithm with elitism. Transportation Research Part C: Emerging Technologies **46** (2014) 30 – 45
10. de Oliveira, R.A., de Medeiros Júnior, M.F., Menezes, R.F.A.: Application of genetic algorithm for optimization on projects of public illumination. Electric Power Systems Research **117** (2014) 84 – 93
11. Uysal, A.K., Gunal, S.: Text classification using genetic algorithm oriented latent semantic features. Expert Systems with Applications **41**(13) (2014) 5938 – 5947
12. Burke, D.S., Jong, K.A.D., Grefenstette, J.J., Ramsey, C.L., Wu, A.S.: Putting more genetics into genetic algorithms. Evolutionary Computation **6**(4) (1998) 387–410
13. Schultz, A.C., Grefenstette, J.J., Jong, K.A.D.: Test and evaluation by genetic algorithms. IEEE Expert **8**(5) (1993) 9–14
14. Jong, K.A.D.: Evolutionary computation - a unified approach. MIT Press (2006)
15. Acevedo-Mosqueda, M.E., Yáñez-Márquez, C., López-Yáñez, I.: Alpha-beta bidirectional associative memories: theory and applications. Neural Processing Letters **26**(1) (2007) 1–40
16. Steinbuch, K.: Adaptive networks using learning matrices. Kybernetik **2**(4) (1964) 148–152

17. Steinbuch, K., Piske, U.A.W.: Learning matrices and their applications. IEEE Trans. Electronic Computers **EC-12**(6) (Dec. 1963) 846–862
18. Kohonen, T.: Correlation matrix memories. IEEE Transactions on Computers **C-21**(4) (1972) 353–359
19. Handels, H., RoSS, T., Kreusch, J., Wolff, H., Pöppl, S.: Feature selection for optimized skin tumor recognition using genetic algorithms. Artificial Intelligence in Medicine **16**(3) (1999) 283 – 297 1999.pdf.
20. Asuncion, A., Newman, D.: UCI Machine Learning Repository (2007)