Evolutionary local improvement on genetic algorithms for feature selection

Leandro D. Vignolo and Matias F. Gerard
Research Institute for Signals, Systems and Computational Intelligence, sınc(i), FICH-UNL/CONICET
Ciudad Universitaria UNL, 4th floor FICH, Santa Fe (3000), Argentina.
Email: {ldvignolo, mgerard}@sinc.unl.edu.ar

Abstract—**Feature selection is an extremely important matter in pattern recognition, particularly when a large set of features is available without knowledge about the discriminative information provided by each element. The key issue is to define a criterion in order to rank the features, discarding those features that are less relevant, redundant, or noisy. This depends on the particular task, the classifier and the properties of the data. A frequent approach consists on the use of genetic algorithms guided by the classification accuracy. However they are often not able to provide a solution with both a considerable reduction of dimensionality and high accuracy rate. Here we propose a modified version of a genetic algorithm, introducing a novel local improvement approach based on evolution, which is able to obtain better dimensionality-accuracy trade-off. Experimental results on different well known datasets show the advantages of our proposal.**

Keywords—**Evolutionary Algorithms; Feature Selection; Local Improvement**

I. Introduction

Many machine learning problems deal with datasets in which every of the given examples consist of a large list of features. In this scenario, feature selection (FS) is essential to perform the classification task with reduced complexity and acceptable performance, specially when prior knowledge of the most discriminant features is not available. Given the complete set of candidate features, FS methods attempt to pick a subset based on one of three criteria: the subset with a specified size that maximizes accuracy; the subset of smaller size that satisfies an accuracy requirement; and the most common case, which is the subset with the best commitment between the dimensionality and the accuracy [1]. In any case, the purpose is the improvement of a machine learning model, either in terms of learning speed, computational complexity, simplicity/interpretability of the representation or generalization capability.

The existing FS techniques can be categorized according to three aspects [2]: the search strategy; the evaluation criterion; and the methodology for the assessment of the performance. The first corresponds to the search strategy which is applied to select candidate solutions among the possible feature combinations. The last two refer to the methods and measures used to evaluate each feature subset, and based on this the FS strategies can be divided in two categories: wrapper methods and filter methods. Wrappers have the advantage that the predictive performance of the selected subset is correlated with the relevance measure. The problem for wrapper methods is that the predictive performance has to be assessed (a classifier has to be trained) for each feature subset tested, being therefore impossible to evaluate all the possible combinations of features. This issue can be overcome using search heuristics like genetic algorithms (GA), particle swarm (PSO) and ant colony optimization (ACO), which are able to find adequate solutions without exploring the entire search space [3]. However, while ACO and PSO typically use encoding based on real numbers, GAs use binary codification, reason why these are more suitable for feature selection problems.

GAs belong to the evolutionary algorithms (EA), a family of biologically inspired techniques that use several mechanisms to imitate natural evolution [4]. GAs and other evolution-based strategies have been successfully used in feature subset optimization problems [5]–[7], and for the searching of optimal representations [8], [9]. Moreover, applications of GA-based wrappers for feature selection include speech recognition [10], face recognition [5] and image analysis [11]. However, classical GA has limitations that are usually neglected in many feature selection applications [12].

Solutions obtained with classical heuristic algorithms are often superior or comparable to those provided by simple GAs. In order to overcome this limitation, one common approach is to hybridize the GA by incorporating domain-specific knowledge. This is accomplished, for example, by using problem-specific encoding or designing appropriate genetic operators [12]. Following these ideas, hybrid GAs have been presented for diverse tasks with advantageous performance [13], [14]. Even though this type of evolution strategies are usually able to gain success, they have to be specially designed for the problem at hand, which makes more difficult or impossible to apply the same FS algorithm to another problem domain.

Other types of evolutionary algorithms with different mechanisms for locally improving the solutions have also been proposed. For example, [15] presents a local improvement procedure based on gradient descent, in order to refine the parameters provided by the EA. This procedure, however, is not appropriate for feature selection. A different approach is presented in [16], where information about the correlation of the features is exploited in order to improve the selection provided by the GA. Even though their results shown improved performances in different datasets, it is not clear how correlation and/or redundant information affects the performance of a classifier, especially when dealing with noisy data. Moreover, studies have shown that redundant features often allow to produce improved results in complex tasks [8], [10].

Here we propose a novel approach to perform local improvement within a GA, applied to feature selection. The strategy is based on the evolution of subordinate populations, which are built every generation based on the chromosomes of the current best individuals in the main population. The subordinate populations are then evolved in a classical GA procedure, and the chromosomes in the main population are replaced if they are improved. Despite that each complete generation in our algorithm requires more evaluations than in a classical GA, it is able to produce better solutions with high accuracy together with lower dimensionality in a reduced number of generations.

The rest of this work is organized as follows. In Section II the classical GA algorithm is first described and then our evolutionary local improvement procedure is explained in detail. The experiments and results are discussed in Section III, and the conclusions together with the future work are commented in Section IV.

II. Materials and Methods

A. Genetic Algorithms

The GAs are the best known meta-heuristic optimization methods. They provide a flexible and robust framework for finding solutions in complex search spaces [17]. The search procedure of GAs is motivated by the laws of natural evolution and it evolves a population of individuals [4]. These individuals consist in chromosomes that carry the information of feasible solutions. The individuals in the population compete with each other in order to produce offspring, and this competition is based on a measure of their aptitude or fitness. This fitness is assessed by evaluating a problem-specific objective function on the information decoded from their chromosomes. This function imitates the selective pressure of the environment.

In a classical GA, the population is subjected to three types of operators: selection, variation (i.e. mutation and crossover) and replacement [18]. The selection operator play as the survival-of-the-fittest mechanism in nature, so that the chance of an individual to survive is proportional to its fitness. On every generation parents are selected from the population, in order to generate the offspring by means of the variation operators. The purpose of the variation operators is to combine information from different individuals (crossover) and also to maintain population diversity, by randomly modifying chromosomes (mutation). A replacement scheme is then applied in order to substitute, completely or partially, the current population by the offspring. These steps are repeated until a desired termination criterion is reached (e.g. a predefined number of generations or a desired fitness value), and then the best individual obtained is taken as the solution for the problem [4]. Another feature of GAs it that they are inherently parallel, and it is relatively straightforward to exploit this in order to increase the computational speed [19]. Algorithm 1, except for steps 9 to 15, represents the pseudocode of a classical GA.

Algorithm 1: GA with evolutionary local improvement

```
1  Initialize the GA population
2  **Evaluate population** (Algorithm 2)
3  repeat
4      Parent selection
5      Mate selected parents (crossover)
6      Mutate offspring
7      Replace population
8      Evaluate population (Algorithm 2)
9      foreach of the N best individuals do
10         Create template chromosome based on selected genes
11         Create subordinate population based on template
12         Evolve subordinate population by classical GA (1 to 8)
13     end
14     Replace chromosomes in the main population
15 until stopping criteria is met
```

Algorithm 2: Evaluate population

```
1  for each individual in the population do
2      Determine feature subset based on chromosome
3      Re-parametrize training patterns using the feature subset
4      Re-parametrize validation patterns using the feature subset
5      Train the classifier on the training set
6      Test the classifier on the validation set
7      Compute fitness using Eq. (1)
8  end
```

B. Fitness calculation

In this work, the evaluation of the population fitness is performed by a function defined as:

$$f(\mathbf{c}) = \alpha \mathcal{R}_{\mathbf{c}} + (1-\alpha)\left[1 - \frac{|\mathbf{c}_{+}|}{|\mathbf{c}|}\right], \qquad (1)$$

where $\mathcal{R}_{\mathbf{c}}$ is the unweighted average recall (UAR) [20] calculated over the dataset, considering only the features selected in the chromosome $\mathbf{c}$; $|\mathbf{c}|$ is the total number of features; $|\mathbf{c}_{+}|$ is the number of features selected in the chromosome $\mathbf{c}$; and α is a weight parameter in $[0,1]$ which specifies the relative importance of each term in the function. The UAR measure is used in (1) instead of classical recall because the former is insensitive to skewed datasets, meaning that it is not affected by the imbalance between the number of examples in each class. The first term of Eq. (1) is in $[0,1]$ range and evaluates the fraction of the data classified correctly. The second term, also in $[0,1]$, evaluates the proportion of the features which are used to perform the classification. The fitness function will increase as the UAR increases and the number of selected features decreases. The steps involved in the computation of the fitness value for each individual are summarized in Algorithm 2.

C. Evolutionary local improvement

The modifications proposed in our approach, called ELIGA (Evolutionary Locally Improved Genetic Algorithm), to the

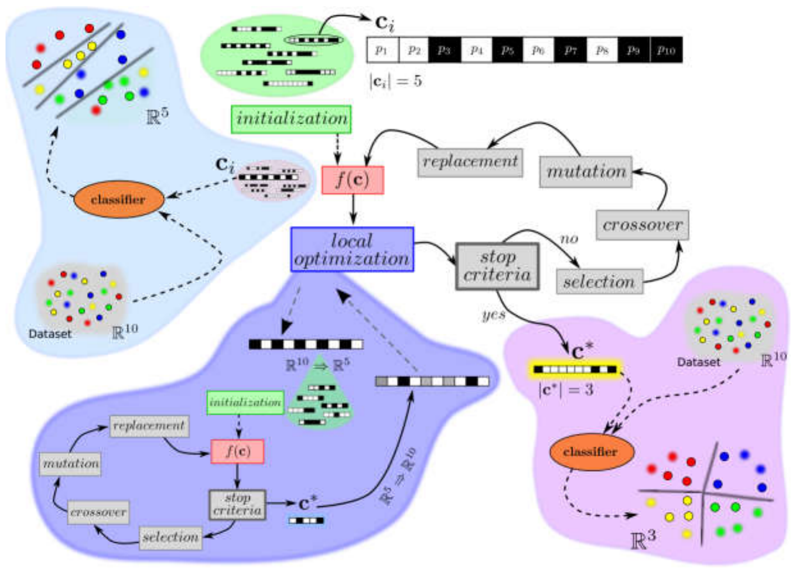

Fig. 1. Outline of the proposed approach for the evolutionary local improvement in genetic algorithms (ELIGA).

classical GA are as follows. Additional steps (9–15) in Algorithm 1 aims to improve best individuals in the current population after the evaluation of the stop criteria. For each selected individual, the improvement steps begin by creating a template chromosome containing only the active features of this individual (line 10). A subordinate population is then created based on this template, and new chromosomes are built as random combinations of these features (line 11). Taking these individuals as the initial population, a classical GA is run and the fitness is evaluated in the same way that in the main population (line 12). Here, the fitness evaluation also involves the classification of a dataset, but taking into account only the features selected in the chromosome (see Algorithm 2). Thus, subordinate populations are evolved in a similar way to the main population, and the number of generations in which these are evolved is additional to the number of generations for the main population. Once the stop criteria is met, a replacement strategy is applied in order to update the main population (line 14). We propose two replacement strategies: parent replacement (ELIGA-PR) and complete replacement (ELIGA-CR). In the parent replacement strategy, the fittest individual for each subordinate population is compared with the parent template, which is replaced in the main population if its performance is improved. In the complete replacement strategy, all the individuals from the main population and from all the subordinate populations are gathered together and sorted according to their fitness. From these, the best N_p individuals are kept to compose the new main population, being N_p the size of the original main population.

Fig. 1 shows an example of a classical GA with the modification proposed. As it can be seen, the main GA runs over the full feature space, and a subordinate GA perform the local optimization of the best individuals (purple area in the figure). In this example, the dataset contain 10 features (p_1 to p_{10}), and chromosomes represent different combinations of them. Once the main GA start, the local optimization step takes the current best individual of the main population and creates a subordinate population to explore the corresponding subset of features (i.e. 5 selected features). Thus, the subordinate population will contain chromosomes with different combinations of those features. Then, a subordinate GA is run on this new features space, evaluating the population with the same fitness function and dataset as in the main GA (cyan color area of the figure). After the subordinate population is evolved, its best individual (2 features or black boxes) is compared with the parent chromosome of the subordinate GA. The later is replaced in the main population if its fitness is improved. When the stop criteria is met, the best set of features found by the algorithm (3 features or black boxes) is evaluated over a separate partition of the dataset, containing only unseen data, to assess the quality of the selected feature set (pink color area of the figure).

D. Decreasing mutation rate

Previous works have shown that the adaptation of the mutation rate within the GA during the evolution promotes the convergence [21]. Particularly, the use of deterministic functions to decrease the mutation rate over time contributes positively in the search [21]. Then we proposed an exponential decay function to update the mutation rate p_m^G on every generation as:

$$p_m^G = \frac{\gamma_1}{|\mathbf{c}|}\left(\frac{\gamma_2}{\gamma_1}\right)^{\frac{G}{G_M}}, \qquad (2)$$

where $|\mathbf{c}|$ is the size of the chromosomes, G is the current generation, G_M is the maximum number of generations, and γ_1 and γ_2 are the number of genes to randomly mutate at the beginning and the end of the search for a given

TABLE I
SUMMARY OF THE DATASETS USED IN THE EXPERIMENTS.

	Attributes	Train	Validation	Test	Categories
Madelon	500	1400	600	600	2
Leukemia	7129	28	18	34	2
Dexter	20000	300	300	2000	2
GCM	16063	72	72	46	14

a chromosome, respectively. This mutation rate is used to evaluate the application of the mutation operator for each gene in the chromosome. For example, specifying $\gamma_1 = 10$ and $\gamma_2 = 1$ would make that 10 genes should be randomly mutated on each chromosome when $G \approx 1$, while only 1 gene should be mutated when $G \approx G_M$. Thus, the function proposed in Eq. (2) allows us to control the mutation rate in order to favor the exploration in the first generations (high mutation rate) and to allow a fine tuning (low mutation rate) in the promising areas of the search space during the last generations.

III. EXPERIMENTS AND RESULTS

A. Datasets

Experiments were performed using four datasets with high dimensionality and different characteristics, including diverse number of features/attributes, examples and classes/categories. The most relevant information of the datasets is summarized in Table I.

1) Madelon: This is an artificial dataset with 500 features, where the target is a multidimensional XOR with five relevant features [22]. It was created for the NIPS 2003 FS challenge[1], and is available at the UCI Repository[2] [2]. Among the 495 remaining characteristics, 15 correspond to linear combinations of the five relevant ones, and the other 480 are noise features. Madelon is a two-class classification problem with sparse binary input variables. The two classes are balanced, and the data is split into training, validation, and test sets with 1400, 600 and 600 examples, respectively. Validation and test sets are stratified, preserving the classes balance.

2) Leukemia: The analysis of gene expression data obtained from DNA micro-arrays is studied in [23] for classification of types of cancer. They constructed a dataset with 7129 gene-expression measurements in classes ALL (acute lymphocytic leukemia) and AML (acute myelogenous leukemia). The problem is to distinguish between these two variants of leukemia (ALL and AML). The data is originally split into two subsets: a training set and an independent test set. The training set consists of 38 samples (27 ALL and 11 AML) from bone marrow specimens. The test set has 34 samples (20 ALL and 14 AML), prepared under different experimental conditions and including 24 bone marrow and 10 blood sample specimens. In our experiments, from the original training set (38 samples), 18 samples (13 ALL and 5 AML) were separated to compose a validation set to be used during the search. From the remaining data (14 ALL and 6 AML), AML samples were repeated until balancing both classes, obtaining 28 training examples at the end.

3) Dexter: The Dexter dataset was created for the NIPS 2003 FS challenge[1] [2] and is hosted by the UCI Repository[2]. It is a subset of the well-known Reuters text categorization benchmark, which was formatted in the "bag-of-words" representation. There are a total of 20000 attributes, from which 9947 are the original features and other 10053 were drawn at random [2]. The original features are normalized and represent frequencies of occurrence of word stems in text. The data was split into training (300), validation (300), and test (2000) sets, all with the classes balanced.

4) GCM: The GCM dataset was complied in [24] and contains the expression profiles of 198 tumor samples representing 14 common human cancer classes[3]. Here the focus was on 190 tumor samples after excluding 8 metastasis samples, and the preprocessing was done according to [24]. Finally, each array was standardized to mean 0 and variance 1 according to [25]. The data set consists of a total of 190 instances, with 16063 attributes (biomarkers) each, and distributed in 14 imbalanced classes. In our experiments, the data was separated in three data sets: training (72 samples), validation (72 samples) and test (42 samples). In order to balance the classes in the training set, preventing the overfitting of the classifier because the majority class, samples were repeated to obtain the same number for each class. At the end, a total of 168 samples were used for training. Conversely, validation and test sets are stratified, maintaining class imbalance.

B. General parameter settings

Aside from the size of the chromosomes, which has to be set according to the number of features in each dataset, all the other parameters of the algorithms were kept fixed in order to allow the comparison. Adequate values for all the parameters were determined by carrying out some preliminary experiments. Appropriate kernel parameters for the SVM classifier were identified performing a grid search on the GCM dataset. For GA and ELIGA, parameters were determined by comparing evolution performance and computation time under different settings. The size of the main population was set to 50 individuals, which were randomly initialized. Chromosomes encode solutions (features subsets) using binary codification, where 1 indicates an active feature. Since we prefer small features subsets, we only allow 3% of the genes in the population to be active in the initialization. For replacing the population we considered an elitist strategy in which the current best individual was maintained, plus a generational gap of 10 individuals which are selected and maintained as well. For the selection of individuals to create the offspring and the generational gap we used the well-known tournament method [17]. The crossover rate was set to 0.9, and the mutation rate was updated as described in Eq. (2) using $\gamma_1 = 10$ and $\gamma_2 = 0.1$ for both, our approach and the classical GA. As finalization criteria, we considered a maximum of 500 generations for all the experiments in both GA and ELIGA.

[1]http://clopinet.com/isabelle/Projects/NIPS2003/
[2]http://archive.ics.uci.edu/ml/datasets.html
[3]Data available online: http://www-genome.wi.mit.edu/MPR/GCM.html

TABLE II
SUMMARY OF THE RESULTS OBTAINED FOR EACH DATASET WITH GA AND ELIGA (MEAN VALUES AND CONFIDENCE INTERVALS OF 95%).

		GA		ELIGA-PR								ELIGA-CR							
				1 subpop		**3 subpop**		**5 subpop**		**10 subpop**		**1 subpop**		**3 subpop**		**5 subpop**		**10 subpop**	
		Mean	CI	Mean	CI	Mean	CI	Mean	CI	Mean	CI	Mean	CI	Mean	CI	Mean	CI	Mean	CI
Madelon	N° Gen.	309.4	±44.6	344.3	±61.1	293.1	±71.3	343.1	±81.6	299.3	±76.0	340.8	±86.1	263.9	±78.7	227.0	±37.5	**206.1**	±73.0
	N° Feat.	61.1	±8.6	34.0	±8.9	27.2	±6.8	25.3	±4.2	22.0	±4.0	20.0	±6.8	**16.2**	±3.0	18.9	±4.3	18.8	±5.1
	Time [s]	**1583**	±645.7	2131	±715.4	4264	±2143.3	6843	±2719.3	7383	±2477.3	1755	±858.2	2154	±1148.6	2433	±809.6	3965	±1216.7
	UAR	0.60	±0.007	0.60	±0.013	**0.61**	±0.004	**0.61**	±0.009	**0.61**	±0.006	**0.61**	±0.008	**0.61**	±0.007	**0.61**	±0.007	**0.61**	±0.005
Leukemia	N° Gen.	488.6	±5.9	80.9	±9.4	81.8	±12.5	84.5	±18.7	86.8	±19.4	30.2	±9.7	28.2	±12.8	**26.6**	±8.7	61.4	±21.5
	N° Feat.	130.9	±5.0	3.8	±0.7	3.0	±0.6	3.0	±0.6	2.2	±0.3	5.3	±1.0	3.4	±0.7	2.8	±0.5	**1.8**	±0.3
	Time [s]	12.3	±0.5	2.4	±0.4	6.3	±1.2	10.4	±4.8	14.5	±5.4	**1.3**	±0.6	2.3	±1.9	3.4	±1.5	13.3	±4.1
	UAR	**0.83**	±0.033	0.63	±0.079	0.66	±0.058	0.70	±0.062	0.64	±0.076	0.65	±0.086	0.63	±0.098	0.68	±0.049	0.62	±0.107
Dexter	N° Gen.	487.7	±3.7	488.7	±8.8	489.7	±12.0	487.3	±7.6	484.7	±6.9	483.3	±9.9	485.5	±7.7	**481.0**	±14.2	487.0	±8.6
	N° Feat.	772.4	±16.0	185.1	±13.1	166.8	±12.0	145.6	±7.4	138.1	±14.1	151.6	±22.3	151.8	±17.8	**135.4**	±10.0	136.0	±12.6
	Time [s]	**1476**	±74.6	2591	±392.8	4251	±694.8	5798	±471.7	11455	±1236.9	1692	±283.9	4150	±632.7	5552	±1040.6	10475	±2048.9
	UAR	0.50	±0.010	0.51	±0.014	0.51	±0.016	0.48	±0.029	0.51	±0.025	**0.52**	±0.018	0.46	±0.052	0.49	±0.025	0.51	±0.026
GCM	N° Gen.	486.6	±6.5	484.1	±7.3	468.8	±18.6	465.1	±33.1	485.1	±15.4	448.5	±30.4	462.4	±31.4	450.3	±38.2	**415.9**	±70.3
	N° Feat.	586.8	±18.8	119.4	±30.4	100.4	±16.5	93.8	±11.9	97.8	±10.6	106.5	±13.9	98.7	±8.4	93.3	±15.6	**91.6**	±15.2
	Time [s]	**183.9**	±4.8	325.7	±60.2	702.8	±130.2	1136.8	±145.6	2274.3	±363.8	318.0	±50.5	596.7	±171.7	817.5	±238.6	1452.7	±495.1
	UAR	**0.45**	±0.026	0.38	±0.043	0.39	±0.039	0.39	±0.043	0.43	±0.034	0.42	±0.036	0.39	±0.029	0.43	±0.035	0.42	±0.049

Also, for both algorithms, we set $\alpha = 0.8$ in Eq. (1) to control the importance of accuracy and dimensionality in the fitness function. In the ELIGA, the subordinate populations were of size 30 and they were evolved for 70 generations. It must be noted that the evolution of the main population is stopped until the subordinate population has completed 70 generations. In order to maintain computational complexity as low as possible, subordinate populations were evolved once every five generations of the main population.

To evaluate the accuracy of candidate solutions, we used a classifier based on support vector machines (SVM) [26]. In particular, the C-SVC implementation was chosen, setting cost $C = 10$ and a linear kernel with slope $a = 10$ and intercept $b = 0.1$. This classifier was trained using a *training* data set, and tested on a *validation* set for the assessment of the accuracy of candidate solutions. Once the search for the best feature subset was finished, the generalization performance was evaluated with the same classifier and a separate *test* set.

C. Results and discussion

The performances of GA and ELIGA were compared based on time (as a measure of computational cost), number and percentage of selected features, and accuracy. Statistics over these measures were computed for each dataset as an average of ten runs of each algorithm. In particular, the performance of ELIGA was evaluated using one, three, five and ten subordinate populations (subpops), and both strategies for the replacement of individuals in the main population (ELIGA-PR and ELIGA-CR), as explained in Section II-C.

Table II summarizes the results obtained for each experiment, showing the mean and confidence interval (CI) for each of the analyzed measures. The CIs were calculated according to [27], specifying a significance level $\alpha = 0,05$ and 9 degrees of freedom. The number of generations and time measures in the table correspond to the moment in which the algorithm converged (the best solution remained unchanged until the 500th generation was reached). As it can be seen, in most cases the time needed to perform the feature selection using ELIGA is increased compared to the classical GA. However, investing more time in this step is preferable in most problems, in order to significantly reduce the computational complexity of the subsequent classification step. Furthermore, it could have been expected that ELIGA would take longer than classical GA, since the evolution of subpops add considerable computational load to the algorithm. However, meeting the goal of finding a reduced set of features while maintaining high accuracy worth overlooking the time performance. Since, once the reduced feature set is obtained, all the following computations for the task at hand would be optimized in time and performance. Regarding dimensionality, the percent of selected features is much lower for both versions of ELIGA in all the datasets, being ELIGA-CR the approach that provides the smallest subsets. Moreover, in the case of Madelon dataset, ELIGA-CR maintained approximately only a quarter of the features kept by the GA. For the Dexter dataset, from 20000 features the GA selected in average 772.40, while the ELIGA-CR kept only 135.40. Clearly, a reduced number of final features allows to focus in the important information, favoring the interpretability and understanding of the problem. Comparing both replacement approaches in our proposal, it can be seen that ELIGA-CR is able to provide smaller feature subsets in a shorter time than ELIGA-PR, without disregarding the classification accuracy. This means that the proposed complete replacement strategy improves the convergence of the algorithm towards promising areas of the search space. Furthermore, except for the Leukemia dataset, the UARs obtained with the algorithms are very close, showing that the ELIGA is able to perform better dimensionality reduction without deteriorating the classification task. It is important to note that confidence intervals support the conclusion that improvements obtained are certainly important.

Another interesting point to discuss is the number of subordinate populations to be used in ELIGA. Experiments show that computational cost is significantly increased with the growth of the number of subordinate populations, though the quality of the solutions obtained is also notably increased. Then, as usual, a trade-off arises regarding the computational

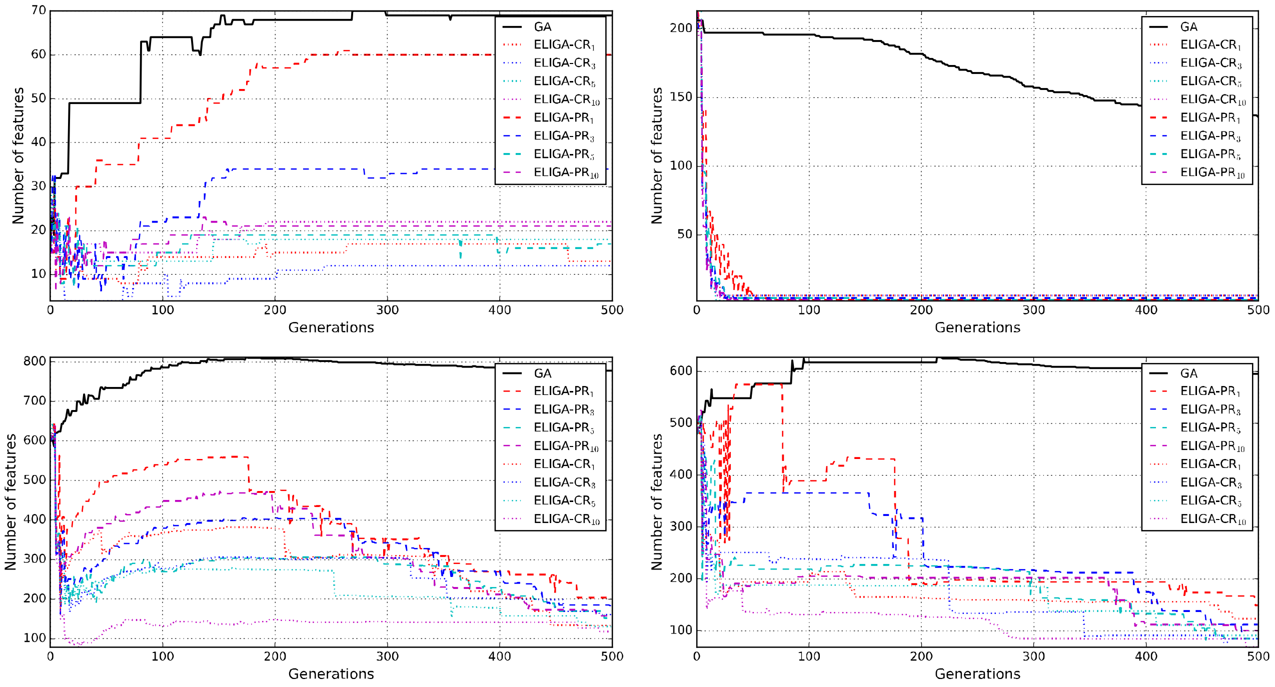

Fig. 2. Evolution of the number of selected features during the optimization for each dataset: Madelon (top left), Leukemia (top right), Dexter (bottom left) and GCM (bottom right).

cost of the search and the quality of the solutions expected.

Fig. 2 shows, for each dataset, the evolution of the number of selected features during the optimization process, comparing classical GA (solid black line), ELIGA-PR (dashed line) and ELIGA-CR (dotted line). Evolutions corresponding to different number of subordinate populations for ELIGA-PR and ELIGA-CR are plotted in different colors, and the corresponding number is indicated as subindex in the plot legends. From left to right and top to bottom, the figures correspond to Madelon, Leukemia, Dexter and GCM datasets, respectively. In the case of Dexter dataset (higher number of features), it can be seen that all algorithms tend to increase the number of features in the first 200 generations, when they are supposed to explore the search space and the mutation rate is high. After that point, they begin to reduce the number of features, being this process considerably faster for ELIGA-PR and ELIGA-CR than for GA. In the last generations, the plot for the GA is slightly flat, showing that it is markedly slower for reducing the number of features. Conversely, the plot for both versions of ELIGA show a notably fast trend reducing the number of features. The bottom left plot of Fig. 2, corresponding to the experiments with the GCM dataset, shows a similar situation. The GA seems to reach a point where no more reduction is possible, whereas the ELIGA continues with an important reduction of the number of features. These analysis suggest that even if the GA is executed with a greater number of generations than ELIGA, the later would still be providing a smaller but useful feature set.

The comparison of both replacement methods shown that dimensionality reduction is faster for ELIGA-PR than ELIGA-CR, as seen in the evolution of the number of features. This could be expected since more individuals are moved from the subordinate populations into the main population in ELIGA-CR. This is also reflected in Table II, since ELIGA-CR provide reduced feature sets for most cases, without impact on the classification accuracy. As a consequence, ELIGA-CR also shows lower convergence time than ELIGA-PR for the same number of subordinate populations (Table II), since the SVM classifier used in the fitness function runs faster with lower dimensionalities. In regard to the number of subordinate populations, it clearly has an important impact on the evolution. However, from these experiments, it is not possible to identify an adequate number of subordinate populations that would be the best in general.

The performance of ELIGA was also compared against three well-known filter methods for feature selection. Correlation-based feature selection (CFS) method sorts features according to pairwise correlations [28]. Laplacian Score (LS) is an unsupervised method that ranks features according to the power of each one for preserving locality [29]. This method constructs a nearest neighbor graph in order to model the local geometric structure, and seeks those features that fit in this graph. Fisher filter is another effective and fast method which computes, for each feature, a score calculated as the ratio of interclass separation and intraclass variance [30]. In order to test these three methods with the datasets considered in this study, we used the implementations provided in the Feature Selection Library (FSLib 2016) [31], [32].

Table III presents the classification performance achieved with the feature sets provided by filter methods and ELIGA.

TABLE III
PERFORMANCE COMPARISON BETWEEN ELIGA-CR, USING 10 SPS, AND DIFFERENT FEATURE RANKING METHODS (UAR).

	Selected features	CFS	Fisher	LS	ELIGA-CR$_{10}$
Madelon	19	0.52	0.56	0.55	**0.61**
Leukemia	2	0.50	0.50	0.50	**0.62**
GCM	92	0.38	**0.42**	0.39	**0.42**

Since CFS, Fisher and LS methods do not provide a feature subset, but they rank all the original features, they require an additional step to choose an appropriate number of top-ranked features. In order to make a fair comparison, we selected as many features from the top of the ranking for every filter-based method, as those selected by ELIGA-CR$_{10}$. This variant of ELIGA was chosen since it provides the best balance between the number of features and accuracy. As it can be seen, ELIGA-CR$_{10}$ (where the subscript indicates the number of subordinate populations) is able to provide a significantly improved classification performance in most cases, when the same number of features are selected. Then, our algorithm is also competitive when comparing with more classical, yet effective, feature ranking methods. These ranking methods, because they are not based on evolutionary computing, are significantly faster than ELIGA and we considered it is not necessary to include the time performance in this comparison. However, we consider that investing time in the feature selection step is worthwhile in many situations. Particularly, it takes relevance when the goal is to improve the performance while reducing the computational complexity of the machine learning algorithm that would take the selected features as input.

IV. CONCLUSIONS AND FUTURE WORK

In this work, we proposed a novel local improvement approach based on the evolution of subordinate populations, that improves the performance of several well-known algorithms in feature selection tasks. We proposed and compared two different strategies for the replacement of individuals from the subordinate populations into the main population, and the performance of the approach was evaluated considering different numbers of subordinate populations. Our proposal was compared against a classical GA and different feature ranking methods on several well-known datasets. Results show that the ELIGA has the ability of performing a deeper search in the solution space, exploiting promising areas by means of the local search. This allows the ELIGA to provide significantly smaller feature subsets, specially for datasets with a large number of attributes, without deteriorating the classification performance. Furthermore, the aggregative fitness function proposed has the flexibility of allowing the use of any type of classifier, and takes into account both accuracy and dimensionality for guiding the search towards the solutions of interest.

In future work we would evaluate and compare the performance of ELIGA taking into account different parameters settings, specially those regarding the subordinate populations. Also, we would consider the use of a different and less complex fitness function for the evaluation of the individuals in the subordinate populations, in order to avoid the increase of computational time. Further, we would be interested in considering different and bigger datasets in the experiments, performing comparisons with other algorithms in the state-of-the-art.

ACKNOWLEDGMENT

The authors would like to thank the *National Agency for Scientific and Technological Promotion* (ANPCyT) (with PICT #2014-1442, PICT #2015-2472), *Universidad Nacional del Litoral* (with PACT 2011 #58, CAI+D 2011 #58-511, CAI+D 2011 #58-525, CAI+D 50020150100059LI and CAI+D 50020150100042LI), and the *National Scientific and Technical Research Council* (CONICET), from Argentina, for their support.

REFERENCES

[1] L. C. Molina, L. Belanche, and À. Nebot, "Feature selection algorithms: a survey and experimental evaluation," in *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on.* IEEE, 2002, pp. 306–313.

[2] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, Eds., *Feature Extraction: Foundations and Applications*, ser. Studies in Fuzziness and Soft Computing. Springer, 2006, vol. 207.

[3] S. M. Vieira, L. F. Mendonça, G. J. Farinha, and J. M. Sousa, "Metaheuristics for feature selection: application to sepsis outcome prediction," in *2012 IEEE Congress on Evolutionary Computation.* IEEE, 2012, pp. 1–8.

[4] J. H. Holland, *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* University of Michigan Press, 1975.

[5] L. D. Vignolo, D. H. Milone, and J. Scharcanski, "Feature selection for face recognition based on multi-objective evolutionary wrappers," *Expert Systems with Applications*, vol. 40, no. 13, pp. 5077–5084, 2013.

[6] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu, "Hybrid feature selection by combining filters and wrappers," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8144 – 8150, 2011.

[7] W. Pedrycz and S. S. Ahmad, "Evolutionary feature selection via structure retention," *Expert Systems with Applications*, vol. 39, no. 15, pp. 11 801 – 11 807, 2012.

[8] L. D. Vignolo, H. L. Rufiner, D. H. Milone, and J. C. Goddard, "Evolutionary Splines for Cepstral Filterbank Optimization in Phoneme Classification," *EURASIP Journal on Advances in Signal Proc.*, vol. 2011, pp. 8:1–8:14, 2011.

[9] C. Charbuillet, B. Gas, M. Chetouani, and J. Zarader, "Optimizing feature complementarity by evolution strategy: Application to automatic speaker verification," *Speech Comm.*, vol. 51, no. 9, pp. 724–731, 2009.

[10] L. D. Vignolo, H. L. Rufiner, and D. H. Milone, "Multi-objective optimisation of wavelet features for phoneme recognition," *IET Signal Processing*, vol. 10, no. 6, pp. 685–691, August 2016.

[11] S. Chatterjee and A. Bhattacherjee, "Genetic algorithms for feature selection of image analysis-based quality monitoring model: An application to an iron mine," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 5, pp. 786 – 795, 2011.

[12] I.-S. Oh, J.-S. Lee, and B.-R. Moon, "Hybrid genetic algorithms for feature selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1424–1437, Nov 2004.

[13] T. N. Bui and B. R. Moon, "Genetic algorithm and graph partitioning," *IEEE Transactions on computers*, vol. 45, no. 7, pp. 841–855, 1996.

[14] L. D. Vignolo, S. M. Prasanna, S. Dandapat, H. L. Rufiner, and D. H. Milone, "Feature optimisation for stress recognition in speech," *Pattern Recognition Letters*, vol. 84, pp. 1 – 7, 2016.

[15] F. Fernndez-Navarro, C. Hervs-Martnez, R. Ruiz, and J. C. Riquelme, "Evolutionary generalized radial basis function neural networks for improving prediction accuracy in gene classification using feature selection," *Applied Soft Computing*, vol. 12, no. 6, pp. 1787 – 1800, 2012.

[16] M. M. Kabir, M. Shahjahan, and K. Murase, "Involving new local search in hybrid genetic algorithm for feature selection," in *Int. Conf. on Neural Information Proc.* Springer, 2009, pp. 150–158.

[17] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, no. 2, pp. 95–99, 1988.

[18] H. Youssef, S. M. Sait, and H. Adiche, "Evolutionary algorithms, simulated annealing and tabu search: a comparative study," *Eng. Applications of Artificial Intelligence*, vol. 14, no. 2, pp. 167 – 181, 2001.

[19] K. Tang, K. F. Man, S. Kwong, and Q. He, "Genetic algorithms and their applications," *IEEE Signal Processing*, vol. 13, no. 6, pp. 22–29, 1996.

[20] A. Rosenberg, "Classifying skewed data: Importance weighting to optimize average recall," in *INTERSPEECH 2012*, Portland, USA, 2012.

[21] T. C. Fogarty, "Varying the probability of mutation in the genetic algorithm," in *Proceedings of the third international conference on Genetic algorithms*. Morgan Kaufmann Pub. Inc., 1989, pp. 104–109.

[22] S. Guérif and Y. Bennani, "Dimensionality reduction trough unsupervised features selection," in *International Conference on Engineering Applications of Neural Networks*, 2007.

[23] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri *et al.*, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *science*, vol. 286, no. 5439, pp. 531–537, 1999.

[24] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.-H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov *et al.*, "Multiclass cancer diagnosis using tumor gene expression signatures," *Proceedings of the National Academy of Sciences*, vol. 98, no. 26, pp. 15 149–15 154, 2001.

[25] S. Dudoit, J. Fridlyand, and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data," *Journal of the American statistical association*, vol. 97, no. 457, pp. 77–87, 2002.

[26] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[27] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers*. John Wiley, 2003.

[28] M. A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., 2000, pp. 359–366.

[29] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Advances in neural information processing systems*, 2006, pp. 507–514.

[30] Q. Gu, Z. Li, and J. Han, "Generalized fisher score for feature selection," in *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2011, pp. 266–273.

[31] G. Roffo and S. Melzi, "Ranking to learn: Feature ranking and selection via eigenvector centrality," *New Frontiers in Mining Complex Patterns, Fifth International workshop, nfMCP2016*, 2017.

[32] G. Roffo, S. Melzi, and M. Cristani, "Infinite feature selection," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 4202–4210.