Neurocomputing 97 (2012) 332–343

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Integrating a differential evolution feature weighting scheme into prototype generation

Isaac Triguero [a,*], Joaquín Derrac [a], Salvador García [b], Francisco Herrera [a]

[a] Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain
[b] Department of Computer Science, University of Jaén, 23071 Jaén, Spain

ARTICLE INFO

Article history:
Received 23 November 2011
Received in revised form
13 March 2012
Accepted 1 June 2012
Communicated by M. Bianchini
Available online 1 July 2012

Keywords:
Differential evolution
Prototype generation
Prototype selection
Feature weighting
Nearest neighbor
Classification

ABSTRACT

Prototype generation techniques have arisen as very competitive methods for enhancing the nearest neighbor classifier through data reduction. Within the prototype generation methodology, the methods of adjusting the prototypes' positioning have shown an outstanding performance. Evolutionary algorithms have been used to optimize the positioning of the prototypes with promising results. However, these results can be improved even more if other data reduction techniques, such as prototype selection and feature weighting, are considered.

In this paper, we propose a hybrid evolutionary scheme for data reduction, incorporating a new feature weighting scheme within two different prototype generation methodologies. Specifically, we will focus on a self-adaptive differential evolution algorithm in order to optimize feature weights and the placement of the prototypes. The results are contrasted with nonparametric statistical tests, showing that our proposal outperforms previously proposed methods, thus showing itself to be a suitable tool in the task of enhancing the performance of the nearest neighbor classifier.

1. Introduction

The designing of classifiers can be considered to be one of the most important tasks in machine learning and data mining [1,2]. Most machine learning methods build a model during the learning process, known as eager learning methods [3], but there are some approaches where the algorithm does not need a model. These algorithms are known as lazy learning methods [4].

The Nearest Neighbor (NN) rule [5] is a simple and effective supervised classification technique which belongs to the lazy learning family of methods. NN is a nonparametric classifier, which requires that all training data instances are stored. Unseen cases are classified by finding the class labels of the closest instances to them. The extended version of NN to k neighbors (kNN) is considered one of the most influential data mining algorithms [6] and it has attracted much attention and research in recent years [7,8]. However, NN may have several disadvantages, such as high computational cost, high storage requirement and sensitivity to noise, which can affect its performance. Furthermore, NN makes predictions over existing data and it assumes that input data perfectly delimits the decision boundaries among classes.

Many approaches have been proposed to improve the performance of the NN rule. One way to simultaneously tackle the computational complexity, storage requirements, and sensitivity to noise of NN is based on data reduction [9]. These techniques try to obtain a reduced version of the original training data, with the double objective of removing noisy and irrelevant data. Taking into consideration the feature space, we can highlight Feature Selection (FS) [10–13] and feature generation/extraction [14] as the main techniques. FS consists of choosing a representative subset of features from the original feature space, while feature generation creates new features to describe the data. From the perspective of the instances, data reduction can be divided into Prototype Selection (PS) [15–17] and Prototype Generation (PG) [18,19]. The former process consists of choosing an appropriate subset of the original training data, while the latter can also build new artificial prototypes to better adjust the decision boundaries between classes in NN classification. In this way, PG does not assume that input data perfectly defines the decision boundaries among classes.

Another way to improve the performance of NN is the employment of weighting schemes. Feature Weighting (FW) [20] is a well known technique which consists of assigning a weight to each feature of the domain of the problem to modify the way in which distances between examples are computed [21]. This technique

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: triguero@decsai.ugr.es (I. Triguero), jderrac@decsai.ugr.es (J. Derrac), sglopez@ujaen.es (S. García), herrera@decsai.ugr.es (F. Herrera).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.06.009

can be viewed as a generalization of FS algorithms, allowing us to obtain a soft approximation of the feature relevance degree assigning a real value as a weight, so different features can receive different treatments.

Evolutionary algorithms [22] have been successfully used in different data mining problems [23,24]. Given that PS, PG and FW problems could be seen as combinatorial and optimization problems, evolutionary algorithms have been used to solve them with excellent results [25]. PS can be expressed as a binary space search problem. To the best of our knowledge, memetic algorithms [26] have provided the best evolutionary model proposed for PS, called SSMA [27]. PG is expressed as a continuous space search problem. Evolutionary algorithms for PG are based on the positioning adjustment of prototypes [28–30], which is a suitable methodology to optimize the location of prototypes. Concretely, Differential Evolution (DE) [31,32] and its advanced approaches [33] have been demonstrated as being the most effective positioning adjustment techniques [34]. Regarding FW methods, many successful evolutionary proposals, most of them based on genetic algorithms, have been proposed, applied to the NN algorithm [35].

Typically, positioning adjustment methods [28,36,30] focus on the placement process and they do not take into consideration the selection of the most appropriate number of prototypes per class. Recently, two different approaches have been proposed in order to tackle this problem. First, in [37,38], this problem is addressed by an iterative addition process that determines which classes need more prototypes to be represented. This algorithm is denoted as IPADECS. Secondly, in [34], the algorithm SSMA-DEPG is presented, in which a previous PS stage is applied to provide the appropriate choice of the number of prototypes per class.

In these techniques, the increase in the size of the data set is a crucial problem. It has been addressed in PS and PG by using stratification techniques [39,40]. They split the data set into various parts to make the application of a prototype reduction technique easier, using a mechanism to join the solutions of each part into a global solution.

The aim of this work is to propose a hybrid approach which combines these two PG methodologies with FW to enhance the NN rule addressing its main drawbacks. In both schemes, the most promising feature weights and location of the prototypes are generated by the SFLSDE algorithm [41] acting as an FW and PG method respectively. Evolutionary PG methods usually tend to overfit the training data in a small number of iterations. For this reason we apply, during the evolutionary optimization process, an FW stage to modify the fitness function of the PG method and determine the relevance of each feature.

The hybridization of the PG and FW problems is the main contribution of this paper, which can be divided into three objectives:

- To propose a new FW technique based on a self-adaptive DE. To the best of our knowledge, DE has not yet been applied to the FW problem.
- To carry out an empirical study to analyze the hybridization models in terms of classification accuracy. Specifically, we will analyze whether the integration of an FW stage with PG methods improves the quality of the resulting reduced sets.
- To check the behavior of these hybrid approaches when dealing with huge data sets, developing a stratified model with the proposed hybrid scheme.

To test the behavior of these approaches, the experimental study will include a statistical analysis based on nonparametric statistical tests [42]. We shall conduct experiments involving a total of 46 classification data sets with different properties.

In order to organize this paper, Section 2 describes the background of PS, PG, FW, DE and stratification. Section 3 explains the hybridization algorithms proposed. Section 4 discusses the experimental framework and Section 5 presents the analysis of results. Finally, in Section 6 we summarize our conclusions.

2. Background

This section covers the background information necessary to define and describe our proposals. Section 2.1 presents a formal definition of PS and PG problems. Section 2.2 describes the main characteristic of FW. Section 2.3 explains the DE technique. Finally, Section 2.4 details the characteristics of the stratification procedure.

2.1. PS and PG problems

This section presents the definition and notation for both PS and PG problems.

A formal specification of the PS problem is the following: let $\mathbf{x}_p$ be an example where $\mathbf{x}_p = (\mathbf{x}_{p1}, \mathbf{x}_{p2}, \ldots, \mathbf{x}_{pD}, \omega)$, with $\mathbf{x}_p$ belonging to a class ω given by $\mathbf{x}_{p\omega}$ and a D-dimensional space in which $\mathbf{x}_{pi}$ is the value of the i-th feature of the p-th sample. Then, let us assume that there is a training set TR which consists of $\mathbf{n}$ instances $\mathbf{x}_p$ and a test set TS composed of $\mathbf{t}$ instances $\mathbf{x}_q$, with ω unknown. Let $SS \subseteq TR$ be the subset of selected samples resulting from the execution of a PS algorithm, then we classify a new pattern $\mathbf{x}_q$ from TS by the NN rule acting over SS.

The purpose of PG is to obtain a prototype generated set GS, which consists of $\mathbf{r}$, $\mathbf{r} < \mathbf{n}$, prototypes, which are either selected or generated from the examples of TR. The prototypes of the generated set are determined to efficiently represent the distributions of the classes and to discriminate well when used to classify the training objects. Their cardinality should be sufficiently small to reduce both the storage and evaluation time spent by an NN classifier.

Both methodologies have been widely studied in the specialized literature. More than 50 PS methods have been proposed. In general, they can be categorized into three kinds of methods: condensation [43], edition [44] or hybrid models [27]. A complete review of this topic is proposed in [17]. Regarding PG techniques, they can be divided into several families depending on the main heuristic operation followed: positioning adjustment [30], class re-labeling [45], centroid-based [18] and space-splitting [46]. A recent PG review is proposed in [19].

More information about PS and PG approaches can be found at the SCI2S thematic public website on *Prototype Reduction in Nearest Neighbor Classification: Prototype Selection and Prototype Generation*.[1]

2.2. Feature weighting

The aim of FW methods is to reduce the sensitivity to redundant, irrelevant or noisy features in the NN rule, by modifying its distance function with weights. These modifications allow us to perform more robust classification tasks, increasing in this manner the global accuracy of the classifier.

The most well known distance or dissimilarity measure for the NN rule is the Euclidean Distance (Eq. (1)), where x_p and x_q are two examples and D is their number of features. We will use it throughout this study as it is simple, easy to optimize, and has

[1] http://sci2s.ugr.es/pr/

been widely used in the field of instance based learning [47]

$$EuclideanDistance(X,Y) = \sqrt{\sum_{i=0}^{D}(x_{pi}-x_{qi})^2} \quad (1)$$

FW methods often extend this equation to apply different weights to each feature (W_i) which modify the way in which the distance measure is computed (Eq. (2))

$$FWDist(X,Y) = \sqrt{\sum_{i=0}^{D} W_i \cdot (x_{pi}-x_{qi})^2} \quad (2)$$

This technique has been widely used in the literature. As far as we know, the most complete study performed can be found in [20], in which a review of several FW methods for lazy learning algorithms is presented (with most of them applied to improve the performance of the NN rule). In this review, FW techniques were categorized by several dimensions, according to its weight learning bias, the weight space (binary or continuous), the representation of features, their generality and the degree of employment of domain specific knowledge.

A wide number of FW techniques are available in the literature, both classical (see [20]) and recent (for example, [21,35]). The most well known group of them is the family of *Relief-based* algorithms. The Relief algorithm [48] (which was originally an FS method) has been widely studied and modified, producing some interesting versions of the original approach [49]. Some of them are based on ReliefF [50] which is the first step of development of *Relief-based* methods as FW techniques [51].

Finally, it is important to note that it is possible to find some approaches dealing simultaneously with FW and FS tasks, for instance, inside a Tabu Search procedure [52] or by managing ensemble-based approaches [53].

2.3. Differential evolution

DE follows the general procedure of an evolutionary algorithm [33]. DE starts with a population of *NP* solutions, so-called individuals. The initial population should cover the entire search space as much as possible. In some problems, this is achieved by uniformly randomizing individuals, but in other problems, such as the PG problem, basic knowledge of the problem is available and the use of other initialization mechanisms is more effective. The subsequent generations are denoted by $G = 0, 1, \ldots, G_{max}$. In DE, it is common to denote each individual as a *D*-dimensional vector $X_{i,G} = \{x^1_{i,G}, \ldots, x^D_{i,G}\}$, called a "target vector".

After initialization, DE applies the mutation operator to generate a mutant vector $V_{i,G}$, with respect to each individual $X_{i,G}$, in the current population. For each target $X_{i,G}$, at the generation *G*, its associated mutant vector $V_{i,G} = \{V^1_{i,G}, \ldots, V^D_{i,G}\}$. The method of creating this mutant vector is that which differentiates one DE scheme from another. In this work, we will focus on the *RandToBest/1* which generates the mutant vector as follows:

$$V_{i,G} = X_{i,G} + F \cdot (X_{best,G} - X_{i,G}) + F \cdot (X_{r^i_1,G} - X_{r^i_2,G}) \quad (3)$$

The indices r^i_1, r^i_2 are mutually exclusive integers randomly generated within the range [1,*NP*], which are also different from the base index *i*. The scaling factor *F* is a positive control parameter for scaling the different vectors.

After the mutation phase, the crossover operation is applied to each pair of the target vector $X_{i,G}$ and its corresponding mutant vector $V_{i,G}$ to generate a new trial vector that we denote $U_{i,G}$. We will focus on the binomial crossover scheme, which is performed on each component whenever a randomly picked number between 0 and 1 is less than or equal to the crossover rate (*CR*), which controls the fraction of parameter values copied from the mutant vector. Then, we must decide which individual should survive in the next generation $G+1$. The selection operator is described as follows:

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if } \mathcal{F}(U_{i,G}) \text{ is better than } \mathcal{F}(X_{i,G}) \\ X_{i,G} & \text{Otherwise} \end{cases}$$

where $\mathcal{F}$ is the fitness function to be minimized. If the new trial vector yields a solution equal to or better than the target vector, it replaces the corresponding target vector in the next generation; otherwise the target is retained in the population. Therefore, the population always gets better or retains the same fitness values, but never deteriorates. This one-to-one selection procedure is generally kept fixed in most of the DE algorithms.

The success of DE in solving a specific problem crucially depends on choosing the appropriate mutation strategy and its associated control parameter values (*F* and *CR*) that determine the convergence speed. Hence, a fixed selection of these parameters can produce slow and/or premature convergence depending on the problem. Thus, researchers have investigated the parameter adaptation mechanisms to improve the performance of the basic DE algorithm [54–56].

One of the most successful adaptive DE algorithms is the Scale Factor Local Search in Differential Evolution (SFLSDE) proposed by [41]. This method was established as the best DE technique for PG in [34].

2.4. Stratification for prototype reduction schemes

When performing data reduction, the scaling up problem appears as the number of training examples increases beyond the capacity of the prototype reduction algorithms, harming their effectiveness and efficiency. This is a crucial problem which must be overcome in most practical applications of data reduction methods. In order to avoid it, in this work we will consider the use of the stratification strategy, initially proposed in [39] for PS, and [40] for PG.

This stratification strategy splits the training data into disjoint strata with equal class distribution. The initial data set *D* is divided into two sets, *TR* and *TS*, as usual (for example a 10th of the data for *TS*, and the rest for *TR* in a 10-fold cross validation). Then, *TR* is divided into *t* disjoint sets TR_j, strata of equal size, TR_1, $TR_2 \cdots TR_t$, maintaining class distribution within each subset. In this manner, the subsets *TR* and *TS* can be represented as follows:

$$TR = \bigcup_{j=1}^{t} TR_j, \quad TS = D \backslash TR \quad (4)$$

Then, a prototype reduction method should be applied to each TR_j, obtaining a reduced set RS_j for each partition.

In PS and PG stratification procedures, the final reduced set is obtained joining every RS_j obtained, and it is denoted as Stratified Reduced Set (SRS)

$$SRS = \bigcup_{j=1}^{t} RS_j \quad (5)$$

When the *SRS* has been obtained, it is ready to be used by an NN classifier to classify the instances of *TS*.

The use of the stratification procedure does not have a great cost in time. Usually, the process of splitting the training data into strata, and joining them when the prototype reduction method has been applied, is not time-consuming, as it does not require any kind of additional processing. Thus, the time needed for the stratified execution is almost the same as that taken in the execution of the prototype reduction method in each strata, which is significantly lower than the time spent if no stratification is applied, due to the time complexity of the PS and PG methods, which most of the time is $O(N^2)$ or higher.

The prototypes present in *TR* are independent of each other, so the distribution of the data into strata will not degrade their representation capabilities if the class distribution is maintained. The number of strata, which should be fixed empirically, will determine the size of them. By using a proper number it is possible to greatly reduce the training set size. This situation allows us to avoid the drawbacks that appeared due to the *scaling up* problem.

3. Hybrid evolutionary models integrating feature weighting and prototype generation

In this section we describe in depth the proposed hybrid approaches and their main components. First of all, we present the proposed FW scheme based on DE (Section 3.1). Next, as we established previously, we will design a hybrid model with the proposed FW for each of the two most effective PG methodologies, IPADECS [37] (Section 3.2) and SSMA-DEPG [34] (Section 3.3). Finally, we develop a stratified model for our hybrid proposals (Section 3.4).

3.1. Differential evolution for feature weighting

As we stated before, FW can be viewed as a continuous space search problem in which we want to determine the most appropriate weights for each feature in order to enhance the NN rule. Specifically, we propose the use of a DE procedure to obtain the best weights, which allows a given reduced set to increase the performance of the classification made in *TR*. We denote this algorithm as DEFW.

DEFW starts with a population of *NP* individuals $X_{i,G}$. In order to encode a weight vector in a DE individual, this algorithm uses a real-valued vector containing *D* elements corresponding to *D* attributes, which range in the interval [0,1]. It means that each individual $X_{i,G}$ in the population encodes a complete solution for the FW problem. Following the ideas established in [54,55,41], the initial population should better cover the entire search space as much as possible by uniformly randomizing individuals within the defined range.

After the initialization process, DEFW enters in a loop in which mutation and crossover operators, explained in Section 2.3, guide the optimization of feature weights by generating new trial vectors $U_{i,G}$. After applying these operators, we check if there have been values out of range of $[\theta,1]$. If a computed value is greater than 1, we truncate it to 1. Furthermore, based on [48], if this value is lower than a threshold θ, we consider this feature to be irrelevant, and therefore, it is established at 0. In our experiments, θ has been fixed empirically to 0.2.

Finally, the selection operator must decide which generated trial vectors should survive in the population of the next generation $G+1$. For our purpose, the NN rule guides this operator. The instances in *TR* are classified with the prototypes of the reduced set given, but in this case, the distance measure for the NN rule is modified according to Eq. (2), where the weights W_i are obtained from $X_{i,G}$ and $U_{i,G}$. Their corresponding fitness values are measured as the *accuracy*$(\cdot)$ obtained, which represents the number of successful hits (correct classifications) relative to the total number of classifications. We try to maximize this value, so the selection operator can be viewed as follows:

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if } accuracy(ReducedSet, U_{i,G}) \\ & \quad >= accuracy(ReducedSet, X_{i,G}) \\ X_{i,G} & \text{Otherwise} \end{cases} \tag{6}$$

In case of a tie between the values of accuracy, we select the $U_{i,G}$ in order to give the mutated individual the opportunity to enter the population.

In order to overcome the limitation of the parameters' selection (*F* and *CR*), we use the ideas established in [41] to implement a self-adaptive DE scheme.

3.2. IPADECS-DEFW: hybridization with IPADECS

The IPADECS algorithm [37] follows an iterative prototype adjustment scheme with an incremental approach. At each step, an optimization procedure is used to adjust the position of the prototypes, adding new ones if needed. The aim of this algorithm is to determine the most appropriate number of prototypes per class and adjust their positioning during the evolutionary process. Specifically, IPADECS uses the SFLSDE technique as an optimizer with a complete solution per individual codification. At the end of the process, IPADECS returns the best *GS* found.

The hybrid model which composes IPADECS and DEFW can basically be described as the combination of an IPADECS stage and then a DEFW to determine the best weights. Fig. 1 shows the pseudo-code of this hybrid scheme. The algorithm proceeds as follows:

- Initially, we perform an IPADECS algorithm in which all the features have a relevance degree of 1.0 (Instructions 1–4).
- Then, the algorithm enters in a loop in which we try to find the most appropriate weights and placement of the prototypes:
  - Instruction 6 performs a DE optimization of the feature weights, so that, the best *GS* obtained from the IPADECS algorithm is used to determine the appropriate weights, as we described in Section 3.1. Furthermore, the current weights are inserted as one of the individuals of the FW population. In this way, we ensure that the FW scheme does not degrade the performance of the *GS* obtained with IPADECS, due to the selection operator used in DEFW.
  - Next, Instruction 7 generates a new *GS*, with IPADECS, but in this case, the optimization process takes into consideration the new weights to calculate the distances between prototypes (see Eq. (2)). The underlying idea of this instruction is that IPADECS should generate a different *GS* due to the fact that the distance measure has changed, and therefore, the continuous search space has been modified.

```
1:  Weights [1..$D$] = 1.0
2:  bestWeights [1.. $D$] = Weights [1..$D$]
3:  $GS$ = IPADECS (Weights);
4:  $Accuracy$ = Evaluate With Weights ($GS$, $TR$, Weights)
5:  for $i = 1$ to $MAXITER$ do
6:      newWeights[1..$D$] = DEFW ($GS$, Weights)
7:      $GS_{aux}$ = IPADECS (newWeights)
8:      $Accuracy_{trial}$ = EvaluateWithWeights ($GS_{aux}$, $TR$, newWeights)
9:      if $Accuracy_{trial} > Accuracy$ then
10:         $Accuracy = Accuracy_{trial}$
11:         $GS = GS_{aux}$
12:     end if
13:     Weights = newWeights
14: end for
15: return $GS$, Weights
```

Fig. 1. Hybridization of IPADECS and DEFW.

  - After this process, we have to ensure that the new positioning of prototypes GS_{aux} with its respective weights has reported a successful improvement of the accuracy rate with respect to the previous *GS*. If the computed accuracy of the new GS_{aux} and its respective weights is greater than the best accuracy found, we save GS_{aux} as the current *GS* (Instructions 8–12).
  - Instruction 13 stored the obtained weights and they will be used in the next iteration.
- After a previously fixed number of iterations, the hybrid model returns the best *GS* and its respective best feature weights.

3.3. SSMA-DEPGFW: hybridization with SSMA-DEPG

The SSMA-DEPG approach [34] uses a PS algorithm prior to the adjustment process to initialize a subset of prototypes, finding a promising selection of prototypes per class.

Specifically, the SSMA algorithm is applied [27]. This is a memetic algorithm which makes use of a local search or meme specifically developed for the PS problem. The interweaving of the global and local search phases allows the two to influence each other. The resulting *SS* is inserted as one of the individuals of the population in the SFLSDE algorithm, which, in this case, is acting as a PG method. Next, it performs mutation and crossover operations to generate new trial solutions. Again, the NN rule guides the selection operator, therefore the SSMA-DEPG returns the best location of the prototypes, which increases the classification rate.

Fig. 2 outlines the hybrid model. To hybridize FW with SSMA-DEPG, this method is carried out as follows:

- Firstly, it is necessary to apply an SSMA stage to determine the number of prototypes per class (Instruction 1).
- Next, the rest of the individuals are randomly generated, extracting prototypes from the *TR* and keeping the same structure as the *SS* selected by the PS method, thus they must have the same number of prototypes per class, and the classes must have the same arrangement in the matrix $X_{i,G}$.
- At this stage, we have established the relevance degree of all features to 1.0. Then, Instruction 4 determines the best classification accuracy obtained in the *NP* population.
- After this, our hybrid model enters into a cooperative loop between FW and SFLSDE.
  - The proposed FW method is applied with the best *GS* found up to that moment. Once again, the current weights are inserted as one of the individuals of the FW population (Instruction 6).
  - Then, a new optimization stage is applied to all the individuals of the population, with the obtained weights modifying the distance measure between prototypes.
- Finally, the method returns the best *GS* with its appropriate feature weights, and it is ready to be used as a reference set by the NN classifier.

```
1:  $GS[1]$ = SSMA();
2:  Generate $GS[2..NP]$ randomly
    with the prototypes distribution of $GS[1]$
3:  Weights[$1..D$] = 1.0
4:  Determine the best $GS$
5:  for $i = 1$ to $MAXITER$ do
6:      Weights [$1..D$] = DEFW( $GS[best]$, Weights)
7:      $GS[1..NP]$ = SFLSDE( $GS[1..NP]$, Weights)
8:      Determine the best $GS$
9:  end for
10: return $GS[best]$, Weights
```

Fig. 2. Hybridization of SSMA-DEPG and DEFW.

3.4. A stratified scheme for hybrid FW and PG methods

Since the immediate application of these hybrid methods over huge sets should be avoided due to their computational cost, we propose the use of a stratification procedure to mitigate this drawback, and thus develop a suitable approach to huge problems.

PS and PG stratified models join every resulting set RS_j, obtained as the application of these techniques to each strata TR_j. Nevertheless, in the proposed hybrid scheme, we obtain for each strata a generated reduced set and its respective feature weights. To develop a stratified method, we study two different strategies:

- *Join procedure*: In this variant, the *SRS* is also generated as the sum of each RS_j. However, the weight of each feature is recalculated, applying the DEFW algorithm. In this case, it uses the *SRS* set as a given reduced set. The stratified method returns *SRS* and its obtained weights to classify the instances of *TS*.
- *Voting rule*: This approach consists of applying a majority voting rule. Each strata RS_j and its respective weights are used to calculate the possible class of each instance of *TS*. The final assigned class is produced via majority voting of the computed class per strata. In our implementation, ties are randomly decided.

4. Experimental framework

In this section, we present the main characteristics related to the experimental study. Section 4.1 introduces the data sets used in this study. Section 4.2 summarizes the algorithms used for comparison with their respective parameters. Finally, Section 4.3 describes the statistical tests applied to contrast the results obtained.

4.1. Data sets

In this study, we have selected 40 classification data sets for the main experimental study. These are well-known problems in the area, taken from the KEEL data set repository[2] [57]. Table 1 summarizes the properties of the selected data sets. It shows, for each data set, the number of examples (#Ex.), the number of attributes (#Atts.), and the number of classes (#Cl.). The data sets considered in this study contain between 100 and 20 000 instances, and the number of attributes ranges from 2 to 85. In addition, they are partitioned using the 10 fold cross-validation (10-fcv) procedure and their values are normalized in the interval [0,1] to equalize the influence of attributes with different range domains. In addition, instances with missing values have been discarded before the execution of the methods over the data sets.

Furthermore, we will perform an additional experiment applying our hybrid models to six huge data sets, which contain more than 20 000 instances. Table 2 shows their characteristics, including the exact number of strata (#Strata.) and instances per strata (#Instances/Strata.).

4.2. Comparison algorithms and parameters

In order to perform an exhaustive study of the capabilities of our proposals, we have selected some of the main proposed models in the literature of PS, PG and FW. In addition, the NN rule with $k=1$ (1NN) has been included as a baseline limit of performance. Apart from SSMA, IPADECS and SSMA-DEPG, which

[2] http://sci2s.ugr.es/keel/datasets

Table 1
Summary description for classification data sets.

Data set	#Ex.	#Atts.	#Cl.	Data set	#Ex.	#Atts.	#Cl.
Sbalone	4174	8	28	Lym	148	18	4
Banana	5300	2	2	Magic	19 020	10	2
Bands	539	19	2	Mammographic	961	5	2
Breast	286	9	2	Marketing	8993	13	9
Bupa	345	6	2	Monks	432	6	2
Chess	3196	36	2	Newthyroid	215	5	3
Cleveland	297	13	5	Nursery	12 690	8	5
Coil2000	9822	85	2	Pima	768	8	2
Contraceptive	1473	9	3	Ring	7400	20	2
Crx	125	15	2	Saheart	462	9	2
Dermatology	366	33	6	Spambase	4597	57	2
Flare-solar	1066	9	2	Spectheart	267	44	2
German	1000	20	2	splice	3190	60	3
Glass	214	9	7	Tae	151	5	3
Haberman	306	3	2	Thyroid	7200	21	3
Hayes-roth	133	4	3	Titanic	2201	3	2
Heart	270	13	2	Twonorm	7400	20	2
Housevotes	435	16	2	Wisconsin	683	9	2
Iris	150	4	3	Yeast	1484	8	10
led7digit	500	7	10	Zoo	101	16	7

Table 2
Summary description for huge classification data sets.

Data set	#Ex.	#Atts.	#Cl.	#Strata.	#Instances/strata.
Adult	48 842	14	2	10	4884
Census	299 285	41	2	60	4990
Connect-4	67 557	42	3	14	4826
Fars	100 968	29	8	20	5048
Letter	20 000	16	26	4	5000
Shuttle	58 000	9	7	12	4833

have been explained above, the rest of the methods are described as follows:

- *TSKNN*: A Tabu search based method for simultaneous FS and FW, which encodes in its solutions the current set of features selected (binary codification), the current set of weights assigned to features, and the best value of k found for the kNN classifier. Furthermore, this method uses fuzzy kNN [58] to avoid ties in the classification process [52].
- *ReliefF*: The first *Relief-based* method adapted to perform the FW process [50]. Weights computed in the original Relief algorithm are not binarized to $0,1$. Instead, they are employed as final weights for the kNN classifier. This method was marked as the best *performance-based* FW method in [20].
- *GOCBR*: A genetic algorithm designed for simultaneous PS and FW process in the same chromosome. Weights are represented by binary chains, thus preserving binary codification in the chromosomes. It has been applied successfully to several real-world applications [59].

Many different configurations are established by the authors of each paper for the different techniques. We focus this experimentation on the recommended parameters proposed by their respective authors, assuming that the choice of the values of the parameters was optimally chosen. The configuration parameters, which are common to all problems, are shown in Table 3. In all of the techniques, Euclidean distance is used as a similarity function and those which are stochastic methods have been run three times per partition. Note that the values of the parameters F_l, F_u, *iterSFGSS* and *iterSFHC* remain constant in all the DE optimizations, and are the recommended values established in [41]. Implementations of the algorithms can be found in the KEEL software tool [57].

Table 3
Parameter specification for all the methods used in the experimentation.

Algorithm	*Parameters*
SSMA	Population=30, Evaluations=10 000, Crossover Probability=0.5, Mutation Probability=0.001
SSMA-DEPG	PopulationSFLSDE=40, IterationsSFLSDE=500, iterSFGSS=8, iterSFHC=20, Fl=0.1, Fu=0.9
IPADECS	Population=10, iterations of Basic DE=500, iterSFGSS=8, iterSFHC=20, Fl=0.1, Fu=0.9
TSKNN	Evaluations=10 000, M=10, N=2, P=ceil ($\sqrt{\#Features}$)
ReliefF	K value for contributions=Best in [1,20]
GOCBR	Evaluations=10 000, Population=100, Crossover Probability=0.7, Mutation Probability=0.1
SSMA-DEPGFW	*MAXITER*=20, PopulationSFLSDE=40, IterationsSFLSDE=50 PopulationDEFW=25, IterationsDEFW=200, iterSFGSS=8, iterSFHC=20, Fl=0.1, Fu=0.9
IPADECS-DEFW	*MAXITER*=20, PopulationIPADECS=10, iterations of Basic DE=50 PopulationDEFW=25, IterationsDEFW=200, iterSFGSS=8, iterSFHC=20, Fl=0.1, Fu=0.9

4.3. Statistical tools for analysis

Hypothesis testing techniques provide us with a way to statistically support the results obtained in the experimental study, identifying the most relevant differences found between the methods [60]. To this end, the use of nonparametric tests will be preferred over parametric ones, since the initial conditions that guarantee the reliability of the latter may not be satisfied, causing the statistical analysis to lose credibility.

We will focus on the use of the Friedman Aligned-ranks (FA) test [42], as a tool for contrasting the behavior of each of our proposals. Its application will allow us to highlight the existence of significant differences between methods. Later, post hoc procedures like Holm's or Finner's will find out which algorithms are distinctive among the $1*n$ comparisons performed. Furthermore, we will use the Wilcoxon Signed-Ranks test in those cases in which we analyze differences between pairs of methods not marked as significant by the previous tests.

More information about these tests and other statistical procedures specifically designed for use in the field of Machine Learning can be found at the SCI2S thematic public website on *Statistical Inference in Computational Intelligence and Data Mining*.[3]

5. Analysis of results

In this section, we analyze the results obtained from different experimental studies. Specifically, our aims are:

- To compare the proposed hybrid schemes to each other over the 40 data sets (Section 5.1).
- To test the performance of these models in comparison with previously proposed methods (Section 5.2).
- To check if the performance of hybrid models is maintained with huge data sets using the proposed stratified model (Section 5.3).

5.1. Comparison of the proposed hybrid schemes

We focus this experiment on comparing both hybrid schemes in terms of accuracy and reduction capabilities. Fig. 3 shows a star plot in which the obtained accuracy test of IPADECS-DEFW and SSMA-DEPGFW is presented for each data set, allowing us to see in an easier way how both algorithms behave in the same domains.

[3] http://sci2s.ugr.es/sicidm/

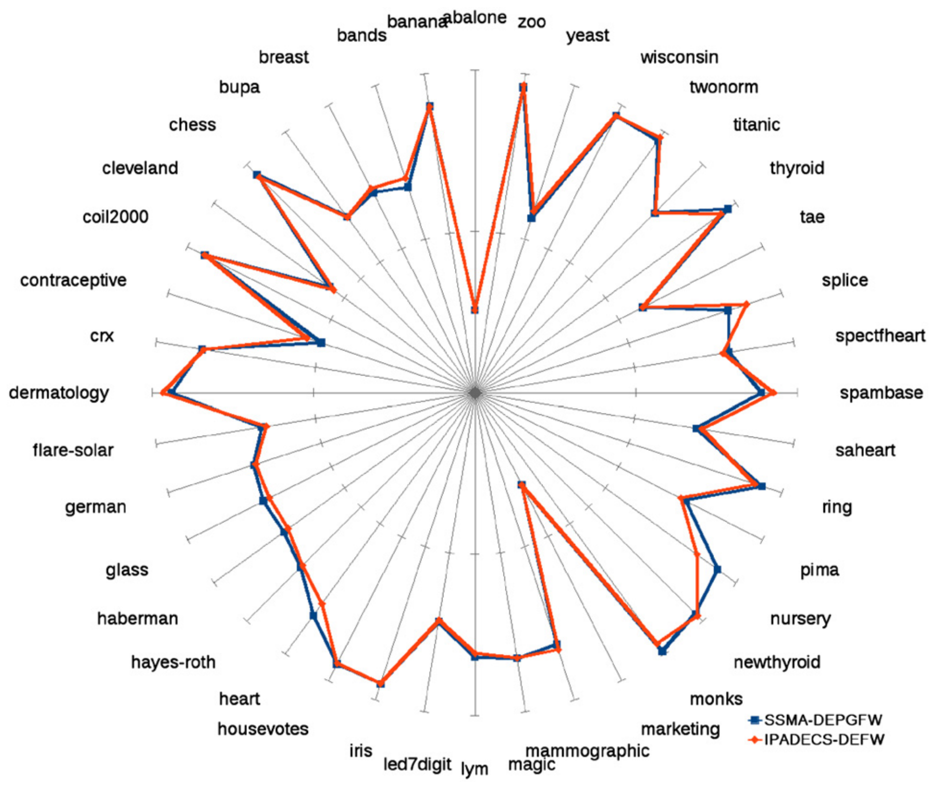

Fig. 3. Accuracy rate comparison.

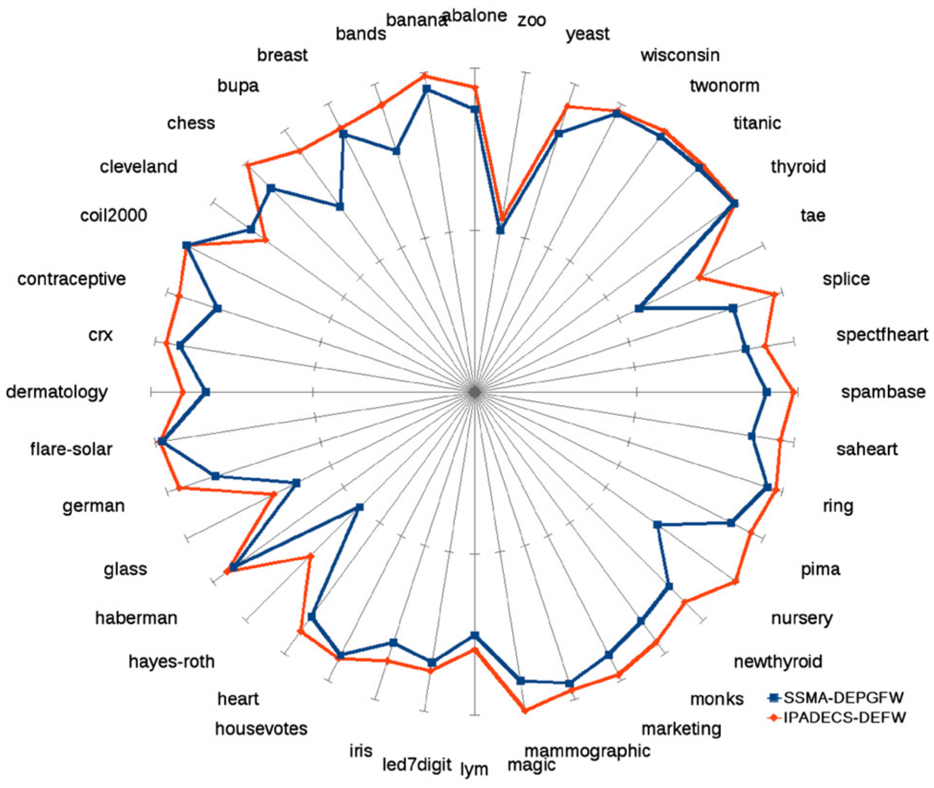

Fig. 4. Reduction rate comparison.

The reduction rate is defined as

$$Reduction\ Rate = 1 - size(GS)/size(TR) \tag{7}$$

It has a strong influence on the efficiency of the solutions obtained, due to the cost of the final classification process performed by the 1NN classifier. Fig. 4 illustrates a star plot representing the reduction rate obtained in each data set for both hybrid models. These star plots represent the performance as the distance from the center; hence a higher area determines the best average performance. The plots allow us to visualize the

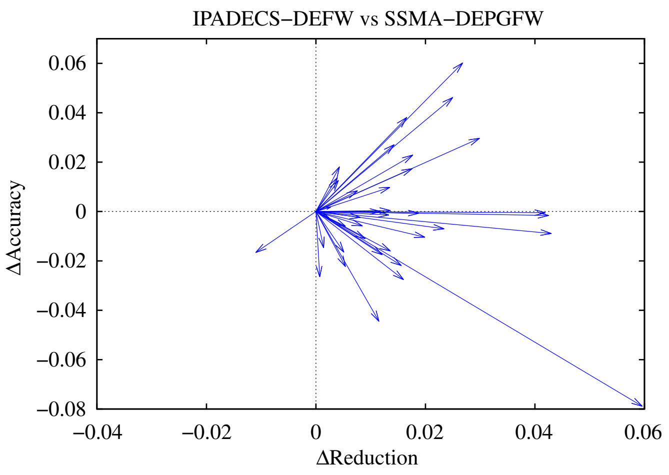

Fig. 5. Accuracy/reduction rates comparison.

Table 4
Results of the Wilcoxon signed-ranks test comparing hybrid schemes.

Comparison	$R+$	$R-$	p-Value
Accuracy rate			
SSMA-DEPGFW vs IPADECS-DEFW	442	338	0.4639
Reduction rate			
IPADECS-DEFW Vs SSMA-DEPGFW	802	18	$\mathbf{4.602 \times 10^{-10}}$

performance of the algorithms comparatively for each problem and in general.

Fig. 5 shows a graphical comparison between these methods considering both objectives simultaneously (accuracy and reduction rate), by using a relative movement diagram [61]. The idea of this diagram is to represent with an arrow the results of two methods on each data set. The arrow starts at the coordinate origin and the coordinates of the tip of the arrow are given by the difference between the reduction (x-axis) and accuracy (y-axis) of IPADECS-DEFW and SSMA-DEPGFW, in this order. Furthermore, numerical results will be presented later in Tables 5 and 6.

Apart from these figures, we use the Wilcoxon test to statistically compare our proposals in both measures. Table 4 collects the results of its application to the accuracy and reduction rates. This table shows the rankings $R+$ and $R-$ values achieved and its associate p-value.

Observing Figs. 3–5 and Table 4 we want to make some comments:

- Fig. 3 shows that both proposals present similar behavior in many domains. Nevertheless, SSMA-DEPGFW obtains the best average result in 24 of the 40 data sets. The Wilcoxon test confirms this statement, showing that there are no significant differences between both approaches and the $R+$ is greater for SSMA-DEPGFW.
- In terms of reduction capabilities, IPADECS-DEFW is shown to be the best performing hybrid model. As the Wilcoxon test reports, it obtains significant differences with respect to SSMA-DEPGFW.
- In Fig. 5, we observe that most of the arrows point out to the right side of the plot. This means that IPADECS-DEFW obtains a lower reduction rate in the problems addressed. Moreover, there are a similar number of arrows pointing up-right and down-left, depicting that the accuracy of both methods is similar. Hence, we can state that IPADECS-DEFW finds the best trade-off between accuracy and reduction rate.

5.2. Comparison with previously proposed methods

In this subsection we perform a comparison between the two proposed hybrid models and the comparison methods established above. We analyze the results obtained in terms of the accuracy in test data and reduction rate.

Table 5 shows the accuracy test results for each method considered in the study. For each data set, the mean accuracy (Acc) and the standard deviation (SD) are computed. The best result for each column is highlighted in bold. The last row presents the average considering all the data sets.

Table 6 presents the reduction rate achieved. Reduction rates are only shown for those methods which perform a relevant reduction of the instances of the *TR*. In this table we can observe that those methods which are based on SSMA obtain the same average reduction rate. This is due to the fact that SSMA is used to obtain the appropriate number of prototypes per class, determining the reduction capabilities at the beginning of the hybrid models: SSMA-DEPG and SSMA-DEPGFW. IPADECS and IPADECS-DEFW obtains slightly different reduction rates because they use the same PG approach changing the fitness function with weights.

To verify the performance of each of our proposals, we have divided the nonparametric statistical study into two different parts. Firstly, we will compare IPADECS-DEFW and SSMA-DEPGFW with the rest of the comparison methods separately (excluding the other proposal) in terms of test accuracy.

Tables 7 and 8 present the results of the FA test for IPADECS-DEFW and SSMA-DEPGFW respectively. In these tables, the computed FA rankings, which represent the associated effectiveness, are presented in the second column. Both tables are ordered from the best (lowest) to the worst (highest) ranking. The third column shows the adjusted p-value (APV) with Holm's test. Finally, the fourth column presents the APV with Finner's test. Note that IPADECS-DEFW and SSMA-DEPGFW are established as control algorithms because they have obtained the best FA ranking in their respective studies. Those APVs highlighted in bold are methods outperformed by the control, at an $\alpha = 0.1$ level of significance.

In this study, we have observed that hybrid schemes perform well with large data sets (those data sets that have more than 2000 instances). We select large data sets, from Table 1, and we compare weighted and unweighted proposals. Fig. 6 shows this comparison. The x-axis position of the point is the accuracy of the original proposal on a single data set, and the y-axis position is the accuracy of the weighted algorithm. Therefore, points above the $y=x$ line correspond to data sets for which new proposals perform better than the original algorithm.

Given Fig. 6 and the results shown before, we can make the following analysis:

- SSMA-DEPGFW and IPADECS-DEFW achieve the best average results. It is important to note that the two hybrid models clearly outperform the methods upon which they are based. The good synergy between PG and FW methods is demonstrated with the obtained results. Specifically, if we focus our attention on those data sets with a large number of features (see splice, chess, etc.), we can state that, in general, the hybridization between PG and a DEFW scheme can be useful to increase the classification accuracy obtained.
- Furthermore, Fig. 6 shows that the proposed weighted algorithms are able to overcome, in most cases, the original proposal when dealing with large data sets.
- Both SSMA-DEPGFW and IPADECS-DEFW achieve the lowest (best) ranking in the comparison. The p-value of the FA test is

Table 5
Accuracy test obtained.

Data sets	1NN		SSMA		SSMA-DEPG		SSMA-DEPGFW		IPADECS		IPADECS-DEFW		TSKNN		ReliefF		GOCBR	
	Acc	SD	Acc	SD	Acc	SD	Acc	SD	Acc	SD	Acc	SD	Acc	SD	Acc	SD	Acc	SD
Abalone	19.91	1.60	**26.09**	1.41	25.66	1.71	25.61	1.34	22.21	2.34	25.47	2.39	24.65	1.43	14.71	1.85	20.75	1.32
Banana	87.51	1.03	89.64	0.89	89.55	1.14	**89.94**	1.16	84.09	4.38	89.70	0.97	89.51	0.84	68.53	2.76	87.87	0.87
Bands	63.09	4.65	59.02	8.98	69.78	6.08	67.00	6.55	67.15	5.91	69.97	5.89	**73.67**	8.33	70.15	6.38	71.45	6.15
Breast	65.35	6.07	**73.79**	4.05	70.32	7.51	69.63	7.64	70.91	7.15	71.00	8.52	72.02	6.45	62.47	9.71	67.14	8.14
Bupa	61.08	6.88	62.79	8.47	66.00	7.80	**67.41**	7.96	65.67	8.48	67.25	5.27	62.44	7.90	56.46	4.37	61.81	6.31
Chess	84.70	2.36	90.05	1.67	90.61	2.18	95.56	1.69	80.22	3.81	94.52	1.03	95.94	0.40	**96.09**	0.57	87.48	1.15
Cleveland	53.14	7.45	54.78	6.29	56.15	6.76	55.80	6.11	52.49	4.48	54.14	6.20	**56.43**	6.84	55.10	8.62	52.80	5.75
Coil2000	89.63	0.77	94.00	0.12	94.00	0.12	94.00	0.12	**94.04**	0.09	94.02	0.09	94.03	0.05	94.02	0.06	91.75	0.40
Contraceptive	42.77	3.69	48.14	5.93	48.74	4.46	50.17	3.35	48.54	4.67	**54.79**	3.61	42.70	0.22	39.99	6.05	43.38	3.65
Crx	79.57	5.12	84.78	4.90	85.65	4.46	85.65	4.83	85.22	4.80	85.07	4.54	**86.23**	3.90	80.43	3.62	84.20	3.91
Dermatology	95.35	3.45	95.10	5.64	95.37	4.04	94.02	4.31	96.18	3.01	**96.73**	2.64	96.47	4.01	95.92	2.77	96.46	2.98
Flare-solar	55.54	3.20	65.47	3.97	66.14	3.42	66.95	3.48	66.23	3.13	65.48	3.25	**67.16**	4.07	57.60	3.51	65.20	3.13
German	70.50	4.25	**73.20**	4.69	71.90	3.11	72.10	5.13	71.80	3.25	71.40	4.27	71.40	2.20	69.30	1.42	70.30	5.37
Glass	73.61	11.91	68.81	8.19	71.98	9.47	73.64	8.86	69.09	11.13	71.45	11.94	76.42	13.21	**80.65**	12.04	67.67	14.10
Haberman	66.97	5.46	73.17	3.75	71.53	6.38	73.18	2.61	**74.45**	6.40	71.53	4.92	74.15	5.07	63.34	8.42	68.94	6.34
Hayes-roth	35.70	9.11	56.18	13.39	75.41	10.57	76.41	10.49	77.05	7.67	75.52	12.11	54.36	11.56	**80.20**	10.67	67.49	10.55
Heart	77.04	8.89	83.70	10.10	82.22	8.25	**85.19**	8.11	83.70	9.83	80.74	9.19	81.48	6.42	78.15	9.72	76.67	8.77
Housevotes	92.16	5.41	92.39	4.99	93.55	5.36	94.24	3.78	92.64	3.71	94.00	4.76	95.16	3.34	94.00	3.48	92.83	6.27
Iris	93.33	5.16	**96.00**	4.42	94.00	4.67	94.67	4.99	94.67	4.00	94.67	4.00	94.00	4.67	94.00	5.54	94.00	3.59
Led7digit	40.20	9.48	34.00	6.69	71.40	4.90	71.80	4.77	**72.40**	3.88	71.20	4.66	10.80	3.12	63.20	5.53	69.80	4.42
Lym	73.87	8.77	**83.03**	13.95	80.29	15.48	81.76	9.83	78.41	9.31	80.66	14.74	74.54	8.95	70.43	22.52	79.34	9.46
Magic	80.59	0.90	82.03	0.75	82.31	0.65	83.24	0.96	80.23	1.47	83.17	1.01	**83.25**	0.68	76.68	5.46	80.66	0.71
Mammographic	73.68	5.59	81.27	5.32	81.27	5.48	81.86	6.03	79.71	4.41	**83.67**	5.55	82.62	4.76	70.76	4.28	78.67	3.84
Marketing	27.38	1.34	30.87	1.63	31.39	0.70	31.90	1.34	30.69	1.11	**31.94**	1.39	24.05	1.33	26.45	1.91	27.19	1.47
Monks	77.91	5.42	96.79	3.31	95.44	3.21	98.86	1.53	91.20	4.76	96.10	2.48	**100.00**	0.00	**100.00**	0.00	79.21	7.15
Newthyroid	97.23	2.26	96.30	3.48	97.68	2.32	96.73	3.64	**98.18**	3.02	97.71	3.07	93.48	2.95	97.25	4.33	94.87	4.50
Nursery	82.67	0.92	85.58	1.17	85.38	1.09	**92.99**	0.76	64.79	4.58	85.10	1.42	82.67	0.88	78.94	32.09	83.53	1.05
Pima	70.33	3.53	74.23	4.01	74.89	5.81	73.23	5.43	**76.84**	4.67	71.63	7.35	75.53	5.85	70.32	5.65	70.59	4.88
Ring	75.24	0.82	92.86	1.03	**93.49**	1.05	93.45	0.64	89.70	1.03	91.22	0.94	84.23	1.17	73.08	1.11	74.54	0.48
Saheart	64.49	3.99	**71.66**	3.46	70.35	5.10	69.47	4.36	70.36	3.07	71.21	3.37	68.22	11.35	60.83	9.15	66.45	14.20
Spambase	89.45	1.17	88.28	1.72	89.84	0.97	88.69	2.12	90.89	0.95	92.50	1.38	**92.54**	1.21	60.58	0.08	89.82	1.48
Spectfheart	69.70	6.55	74.20	8.69	79.02	7.31	79.68	10.73	**80.54**	4.25	77.93	4.70	76.01	10.12	78.30	11.92	74.99	6.87
Splice	74.95	1.15	73.32	1.63	78.37	4.44	82.51	4.80	79.78	3.99	**88.53**	1.99	71.72	1.72	78.24	1.30	74.20	1.54
Tae	40.50	8.43	53.17	12.66	56.54	15.86	**58.38**	11.80	57.71	11.11	58.33	12.04	30.54	2.56	49.12	3.77	55.00	3.98
Thyroid	92.58	0.81	94.14	0.74	94.58	0.55	**96.93**	2.39	93.99	0.36	94.28	0.58	95.87	0.61	92.57	0.26	92.85	0.73
Titanic	60.75	6.61	73.51	2.47	78.96	2.30	78.83	2.22	78.19	2.92	**79.01**	2.11	77.78	2.79	61.33	7.90	78.83	2.22
Twonorm	94.68	0.73	96.34	0.74	96.92	0.79	96.50	0.72	**97.66**	0.69	97.76	0.72	96.96	0.87	94.65	1.01	94.93	1.25
Wisconsin	95.57	2.59	96.57	2.65	96.14	2.12	96.42	2.23	96.42	1.94	96.28	1.72	96.00	3.61	96.28	2.14	**97.14**	3.30
Yeast	50.47	3.91	57.55	1.66	58.09	2.14	56.88	1.60	57.35	3.13	**59.17**	3.61	55.86	12.99	51.55	4.97	53.44	6.50
Zoo	92.81	6.57	85.33	9.73	95.33	6.49	95.83	9.72	96.33	8.23	96.67	6.83	66.25	8.07	**96.83**	2.78	96.17	5.16
Average	70.80	20.06	75.20	18.98	77.66	17.24	**78.43**	17.55	76.44	17.46	78.29	17.29	73.68	22.09	72.46	19.78	74.51	17.89

lower than 10^{-5} in both cases, meaning that significant differences have been detected between the methods of the experiment.

- Holm's procedure states that the differences of IPADECS-DEFW over 1NN, ReliefF, GOCBR, TSKNN and SSMA are significant ($\alpha = 0.1$). Finner's procedure goes further, also highlighting the difference over IPADECS (Finner APV = 0.0926).
- In the case of SSMA-DEPGFW the results are similar: the differences over 1NN, ReliefF, GOCBR, TSKNN and SSMA are marked as significant by Holm's test ($\alpha = 0.1$), whereas Finner's also highlights the difference over IPADECS again (Finner APV = 0.0643).

These results suggest that our proposals, SSMA-DEPGFW and IPADECS-DEFW, significantly improve all the comparison methods considered except SSMA-DEPG. The multiple comparison test applied does not detect significant differences between the three best methods. Hence, we will study this last case carefully, applying a pairwise comparison between our proposals and SSMA-DEPG. Specifically, we will focus on the Wilcoxon test, which allows us to have a further insight into the comparison of this method with our proposals. Table 9 shows the results of its application, comparing SSMA-DEFPGW and IPADECS-DEFW with SSMA-DEPG. The results obtained suggest that it is outperformed by the new proposals, at $\alpha = 0.1$ level. Although this result is not as strong as those differences found by Holm's and Finner's procedures, it still supports the existence of a significant improvement of SSMA-DEPGFW and IPADECS-DEFW over SSMA-DEPG.

5.3. *Analyzing scaling up capabilities: a stratified model*

In this study, we select the hybrid model IPADECS-DEFW as the best trade-off between accuracy and reduction rate to implement a stratified model, considering the two strategies explained in Section 3.4. The performance of this method is analyzed by using six huge data sets taken from the KEEL data set repository (see Table 2).

To check the performance of the proposed stratified models, we perform a comparison with the stratified versions of IPADECS and SSMA-DEPG proposed in [40]. Furthermore, 1NN behavior has also been analyzed as a baseline method for this study. For all the techniques, we used the same set up as that used in the former study, and set up the strata size as near as possible to 5000 instances. Table 2 shows the exact number of strata and instances per strata.

Table 10 shows the accuracy test results for each method considered in this study. For each data set, the mean accuracy (Acc) and the standard deviation (SD) are computed. The best result for each column is highlighted in bold. The last row presents the average considering all the huge data sets. Table 11 collects the

Table 6
Reduction rates obtained.

Data sets	SSMA	SSMA-DEPG	SSMA-DEPGFW	IPADECS	IPADECS DEFW
Abalone	0.9749	0.9749	0.9749	0.9886	0.9882
Banana	0.9900	0.9900	0.9900	0.9981	0.9981
Bands	0.9567	0.9567	0.9567	0.9872	0.9866
Breast	0.9790	0.9790	0.9790	0.9820	0.9829
Bupa	0.9417	0.9417	0.9417	0.9848	0.9842
Chess	0.9782	0.9782	0.9782	0.9981	0.9981
Cleveland	0.9710	0.9710	0.9710	0.9600	0.9600
Coil2000	0.9999	0.9999	0.9999	0.9997	0.9997
Contraceptive	0.9672	0.9672	0.9672	0.9926	0.9922
Crx	0.9844	0.9844	0.9844	0.9929	0.9929
Dermatology	0.9663	0.9663	0.9663	0.9806	0.9806
Flare-solar	0.9955	0.9955	0.9955	0.9969	0.9969
German	0.9686	0.9686	0.9686	0.9940	0.9920
Glass	0.9237	0.9237	0.9237	0.9393	0.9393
Haberman	0.9840	0.9840	0.9840	0.9904	0.9891
Hayes-roth	0.9006	0.9006	0.9006	0.9436	0.9436
Heart	0.9716	0.9716	0.9716	0.9853	0.9831
Housevotes	0.9826	0.9826	0.9826	0.9849	0.9849
Iris	0.9630	0.9630	0.9630	0.9748	0.9748
Led7digit	0.9693	0.9693	0.9693	0.9747	0.9747
Lym	0.9504	0.9504	0.9504	0.9594	0.9594
Magic	0.9808	0.9808	0.9808	0.9996	0.9996
Mammographic	0.9895	0.9895	0.9895	0.9938	0.9938
Marketing	0.9825	0.9825	0.9825	0.9961	0.9961
Monks	0.9750	0.9750	0.9750	0.9910	0.9910
Newthyroid	0.9700	0.9700	0.9700	0.9835	0.9835
Nursery	0.9396	0.9396	0.9396	0.9992	0.9992
Pima	0.9780	0.9780	0.9780	0.9916	0.9916
Ring	0.9902	0.9902	0.9902	0.9956	0.9956
Saheart	0.9735	0.9735	0.9735	0.9931	0.9911
Spambase	0.9805	0.9805	0.9805	0.9971	0.9971
Spectfheart	0.9696	0.9696	0.9696	0.9817	0.9817
Splice	0.9679	0.9679	0.9679	0.9947	0.9947
Tae	0.9139	0.9139	0.9139	0.9558	0.9558
Thyroid	0.9982	0.9982	0.9982	0.9992	0.9989
Titanic	0.9960	0.9960	0.9960	0.9990	0.9987
Twonorm	0.9952	0.9952	0.9952	0.9993	0.9993
Wisconsin	0.9932	0.9932	0.9932	0.9951	0.9951
Yeast	0.9681	0.9681	0.9681	0.9858	0.9858
Zoo	0.9010	0.9010	0.9010	0.9086	0.9086
Average	0.9695	0.9695	0.9695	**0.9842**	0.9840

Table 7
Average FA rankings of IPADECS-DEFW and the rest of the comparison methods.

Algorithm	FA ranking	Holm APV	Finner APV
IPADECS-DEFW	97.6750	–	–
SSMA-DEPG	106.8875	0.6561	0.6561
IPADECS	133.9000	0.1599	**0.0926**
SSMA	149.0250	**0.0392**	**0.0182**
TSKNN	153.1375	**0.0294**	**0.0128**
GOCBR	190.4875	**0**	**0**
ReliefF	209.0500	**0**	**0**
1NN	243.8375	**0**	**0**

p-Value by the FA test $= \mathbf{9.915} \times 10^{-6}$.

reduction rate achieved for each method. In this table, both IPADECS-DEFW variants obtain the same average reduction rate.

In Table 10, we observe that the IPADECS-DEFW model with the join procedure has obtained the best average accuracy result. The Wilcoxon test has been conducted comparing this method with the rest. Table 12 shows the results of its application.

As in the former study, IPADECS-DEFW obtains a slightly lower reduction power than IPADECS as we can see in Table 11.

Observing these tables, we can summarize that with an appropriate stratification procedure, the idea of combining PG and FW is also applicable to huge data sets, obtaining good

Table 8
Average FA rankings of SSMA-DEPGFW and the rest of the comparison methods.

Algorithm	FA ranking	Holm APV	Finner APV
SSMA-DEPGFW	93.8875	–	–
SSMA-DEPG	108.0750	0.4929	0.4929
IPADECS	133.5250	0.1107	**0.0643**
SSMA	149.5250	**0.0215**	**0.0100**
TSKNN	155.2250	**0.0121**	**0.0053**
GOCBR	190.8000	**0**	**0**
ReliefF	208.7250	**0**	**0**
1NN	244.2375	**0**	**0**

p-Value by the FA test $= \mathbf{9.644} \times 10^{-6}$.

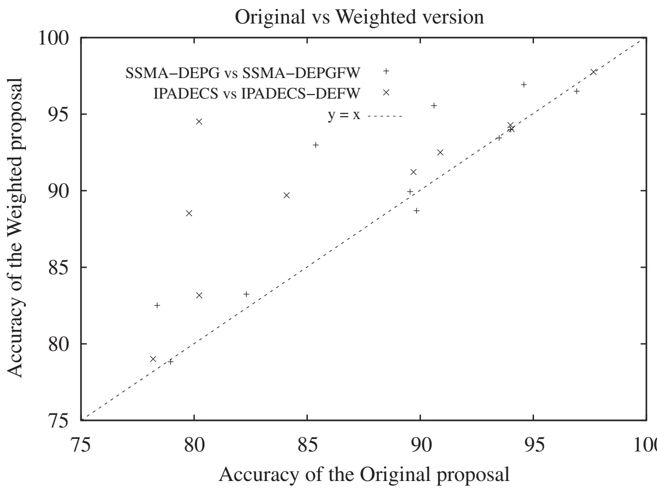

Fig. 6. Accuracy results over large data sets.

Table 9
Results of the Wilcoxon signed-ranks test.

Comparison	$R+$	$R-$	*p*-Value
SSMA-DEPGFW vs SSMA-DEPG	586.5	233.5	**0.0469**
IPADECS-DEFW vs SSMA-DEPG	521	259	**0.0682**

results. The Wilcoxon test supports this statement, showing that IPADECS-DEFW, with the join procedure, is able to significantly outperform IPADECS, SSMA-DEPG and 1NN to a level of $\alpha = 0.1$.

6. Conclusions

In this paper, we have introduced a novel data reduction technique which exploits the cooperation between FW and PG to improve the classification performance of the NN, storage requirements and its running time. A self-adaptive DE algorithm has been used to optimize feature weights and the positioning of the prototypes for the nearest neighbor algorithm, acting as an FW scheme and a PG method, respectively.

The proposed DEFW scheme has been incorporated within two of the most promising PG methods. These hybrid models are able to overcome isolated PG methods due to the fact that FW changes the way in which distances between prototypes are measured, and therefore the adjustment of prototypes can be more refined. Furthermore, we have proposed a stratified procedure specifically designed to deal with huge data sets.

The wide experimental study performed has allowed us to contrast the behavior of these hybrid models when dealing with a wide variety of data sets with different numbers of instances and

Table 10
Accuracy test results in huge data sets.

Data sets	1NN		IPADECS		SSMA-DEPG		IPADECS-DEFW		IPADECS-DEFW	
	Acc	SD	Acc	SD	Acc	SD	Join		Voting rule	
							Acc	SD	Acc	SD
Adult	0.7960	0.0035	0.8263	0.0032	0.8273	0.0098	**0.8335**	0.0077	0.8313	0.0031
Census	0.9253	0.0010	0.9439	0.0005	0.9460	0.0009	**0.9477**	0.0007	0.9428	0.0300
Connect-4	0.6720	0.0036	0.6569	0.0009	0.6794	0.0061	**0.6847**	0.0058	0.6624	0.0045
Fars	0.7466	0.0034	0.7439	0.0218	0.7625	0.0036	**0.7676**	0.0039	0.7536	0.0033
Letter	0.9592	0.0002	0.9420	0.0082	0.9053	0.0082	0.9632	0.0121	**0.9699**	0.0075
Shuttle	**0.9993**	0.0004	0.9941	0.0021	0.9967	0.0021	0.9967	0.0008	0.9967	0.0015
Average	0.8497	0.0020	0.8512	0.0061	0.8529	0.0051	**0.8656**	0.0052	0.8595	0.0083

Table 11
Reduction rate results in huge data sets.

Data sets	IPADECS	SSMA-DEPG	IPADECS-DEFW	IPADECS-DEFW
			Join	Voting rule
Adult	**0.9986**	0.9882	**0.9986**	**0.9986**
Census	**0.9994**	0.9973	0.9987	0.9987
Connect-4	**0.9990**	0.9822	0.9981	0.9981
Fars	**0.9968**	0.9808	0.9957	0.9957
Letter	**0.9924**	0.9805	0.9901	0.9901
Shuttle	**0.9986**	0.9981	0.9971	0.9971
Average	**0.9975**	0.9878	0.9964	0.9964

Table 12
Results obtained by the Wilcoxon test for algorithm IPADECS-DEFW join.

IPADECS-DEFW join VS	$R+$	$R-$	p-Value
1NN	20.0	1.0	**0.0625**
IPADECS	21.0	0.0	**0.0312**
SSMA-DEPG	15.0	0.0	**0.0625**
IPADECS-DEFW voting rule	12.0	3.0	0.1775

features. The proposed stratified procedure has shown that this technique is useful to tackle the scaling up problem. The results have been compared with several nonparametric statistical procedures, which have supported the conclusions drawn.

As future work, we consider that this methodology could be extended by using different learning algorithms such as support vector machines, decision trees, and so on, following the guidelines given in similar studies for training set selection [62–64].

Acknowledgments

Supported by the Research Projects TIN2011-28488 and TIC-6858. J. Derrac holds an FPU scholarship from the Spanish Ministry of Education and Science.

References

[1] E. Alpaydin, Introduction to Machine Learning, 2nd edition, MIT Press, Cambridge, MA, 2010.
[2] I. Kononenko, M. Kukar, Machine Learning and Data Mining: Introduction to Principles and Algorithms, Horwood Publishing Limited, 2007.
[3] T.M. Mitchell, Machine Learning, McGraw-Hill, 1997.
[4] D.W. Aha (Ed.), Lazy Learning, Springer, 1997.
[5] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 13 (1) (1967) 21–27.
[6] X. Wu, V. Kumar (Eds.), The Top Ten Algorithms in Data Mining, Chapman & Hall/CRC Data Mining and Knowledge Discovery, 2009.
[7] Y. Gao, F. Gao, Edited AdaBoost by weighted kNN, Neurocomputing 73 (16–18) (2010) 3079–3088.
[8] J. Derrac, S. García, F. Herrera, IFS-CoCo: instance and feature selection based on cooperative coevolution with nearest neighbor rule, Pattern Recognition 43 (6) (2010) 2082–2105.
[9] D. Pyle, Data Preparation for Data Mining, The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, 1999.
[10] J.M. Urquiza, I. Rojas, H. Pomares, L.J. Herrera, J. Ortega, A. Prieto, Method for prediction of protein–protein interactions in yeast using genomics/proteomics information and feature selection, Neurocomputing 74 (16) (2011) 2683–2690.
[11] H. Liu, H. Motoda (Eds.), Computational Methods of Feature Selection, Chapman & Hall/CRC Data Mining and Knowledge Discovery Series, Chapman & Hall/CRC, 2007.
[12] J.X. Peng, S. Ferguson, K. Rafferty, P. Kelly, An efficient feature selection method for mobile devices with application to activity recognition, Neurocomputing 74 (17) (2011) 3543–3552.
[13] J. Derrac, C. Cornelis, S. García, F. Herrera, Enhancing evolutionary instance selection algorithms by means of fuzzy rough set based feature selection, Inf. Sci. 186 (1) (2012) 73–92.
[14] H. Liu, H. Motoda, Feature Extraction, Construction and Selection: A Data Mining Perspective, Kluwer Academic Publishers, 2001.
[15] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Mach. Learn. 38 (3) (2000) 257–286.
[16] A. Guillén, L.J. Herrera, G. Rubio, H. Pomares, A. Lendasse, I. Rojas, New method for instance or prototype selection using mutual information in time series prediction, Neurocomputing 73 (10–12) (2010) 2030–2038.
[17] S. García, J. Derrac, J. Cano, F. Herrera, Prototype selection for nearest neighbor classification: taxonomy and empirical study, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 417–435.
[18] H.A. Fayed, S.R. Hashem, A.F. Atiya, Self-generating prototypes for pattern classification, Pattern Recognition 40 (5) (2007) 1498–1509.
[19] I. Triguero, J. Derrac, S. García, F. Herrera, A taxonomy and experimental study on prototype generation for nearest neighbor classification, IEEE Trans. Syst. Man Cybern.—Part C: Appl. Rev. 42 (1) (2012) 86–100.
[20] D. Wettschereck, D.W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artif. Intell. Rev. 11 (1997) 273–314.
[21] R. Paredes, E. Vidal, Learning weighted metrics to minimize nearest-neighbor classification error, IEEE Trans. Pattern Anal. Mach. Intell. 28 (7) (2006) 1100–1110.
[22] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer-Verlag, Berlin, 2003.
[23] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer-Verlag, Berlin, 2002.
[24] G.L. Pappa, A.A. Freitas, Automating the Design of Data Mining Algorithms: An Evolutionary Computation Approach, Natural computing, Springer, 2009.
[25] J.R. Cano, F. Herrera, M. Lozano, Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study, IEEE Trans. Evol. Comput. 7 (6) (2003) 561–575.
[26] N. Krasnogor, J. Smith, A tutorial for competent memetic algorithms: model, taxonomy, and design issues, IEEE Trans. Evol. Comput. 9 (5) (2005) 474–488.
[27] S. García, J.R. Cano, F. Herrera, A memetic algorithm for evolutionary prototype selection: a scaling up approach, Pattern Recognition 41 (8) (2008) 2693–2709.
[28] F. Fernández, P. Isasi, Evolutionary design of nearest prototype classifiers, J. Heuristics 10 (4) (2004) 431–454.
[29] A. Cervantes, I.M. Galván, P. Isasi, AMPSO: a new particle swarm method for nearest neighborhood classification, IEEE Trans. Syst. Man Cybern.—Part B: Cybern. 39 (5) (2009) 1082–1091.
[30] L. Nanni, A. Lumini, Particle swarm optimization for prototype reduction, Neurocomputing 72 (4–6) (2008) 1092–1097.
[31] R. Storn, K.V. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (10) (1997) 341–359.

[32] K.V. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Natural Computing Series, , 2005.
[33] S. Das, P. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31.
[34] I. Triguero, S. García, F. Herrera, Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification, Pattern Recognition 44 (4) (2011) 901–916.
[35] F. Fernández, P. Isasi, Local feature weighting in nearest prototype classification, IEEE Trans. Neural Networks 19 (1) (2008) 40–53.
[36] J. Li, M.T. Manry, C. Yu, D.R. Wilson, Prototype classifier design with pruning, Int. J. Artif. Intell. Tools 14 (1–2) (2005) 261–280.
[37] I. Triguero, S. García, F. Herrera, IPADE: iterative prototype adjustment for nearest neighbor classification, IEEE Trans. Neural Networks 21 (12) (2010) 1984–1990.
[38] I. Triguero, S. García, F. Herrera, Enhancing IPADE algorithm with a different individual codification, in: Proceedings of the 6th International Conference on Hybrid Artificial Intelligence Systems (HAIS), Lecture Notes in Artificial Intelligence, vol. 6679, 2011, pp. 262–270.
[39] J.R. Cano, F. Herrera, M. Lozano, Stratification for scaling up evolutionary prototype selection, Pattern Recognition Lett. 26 (7) (2005) 953–963.
[40] I. Triguero, J. Derrac, S. García, F. Herrera, A study of the scaling up capabilities of stratified prototype generation, in: Proceedings of the Third World Congress on Nature and Biologically Inspired Computing (NABIC'11), 2011, pp. 304–309.
[41] F. Neri, V. Tirronen, Scale factor local search in differential evolution, Memetic Comput. 1 (2) (2009) 153–171.
[42] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inf. Sci. 180 (2010) 2044–2064.
[43] P.E. Hart, The condensed nearest neighbor rule, IEEE Trans. Inf. Theory 18 (1968) 515–516.
[44] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Trans. Syst. Man Cybern. 2 (3) (1972) 408–421.
[45] J.S. Sánchez, R. Barandela, A.I. Marqués, R. Alejo, J. Badenas, Analysis of new techniques to obtain quality training sets, Pattern Recognition Lett. 24 (7) (2003) 1015–1022.
[46] J.S. Sánchez, High training set size reduction by space partitioning and prototype abstraction, Pattern Recognition 37 (7) (2004) 1561–1564.
[47] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Mach. Learn. 6 (1) (1991) 37–66.
[48] K. Kira, L.A. Rendell, A practical approach to feature selection, in: Proceedings of the Ninth International Conference on Machine Learning, Morgan Kaufmann, Aberdeen, Scotland, 1992, pp. 249–256.
[49] K. Ye, K. Feenstra, J. Heringa, A. Ijzerman, E. Marchiori, Multi-RELIEF: a method to recognize specificity determining residues from multiple sequence alignments using a machine learning approach for feature weighting, Bioinformatics 24 (1) (2008) 18–25.
[50] I. Kononenko, Estimating attributes: analysis and extensions of RELIEF, in: Proceedings of the 1994 European Conference on Machine Learning, Springer Verlag, Catania, Italy, 1994, pp. 171–182.
[51] M.R. Sikonja, I. Kononenko, Theoretical and empirical analysis of ReliefF and RReliefF, Mach. Learn. 53 (1-2) (2003) 23–69.
[52] M.A. Tahir, A. Bouridane, F. Kurugollu, Simultaneous feature selection and feature weighting using hybrid tabu search/k-nearest neighbor classifier, Pattern Recognition Lett. 28 (4) (2007) 438–446.
[53] J. Gertheiss, G. Tutz, Feature selection and weighting by nearest neighbor ensembles, Chemometr. Intell. Lab. Syst. 99 (2009) 30–38.
[54] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[55] S. Das, A. Abraham, U.K. Chakraborty, A. Konar, Differential evolution using a neighborhood-based mutation operator, IEEE Trans. Evol. Comput. 13 (3) (2009) 526–553.
[56] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[57] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, J. Mult. 17 (2–3) (2011) 255–287.
[58] J.M. Keller, M.R. Gray, J.A. Givens, A fuzzy K-nearest neighbor algorithm, IEEE Trans. Syst. Man Cybern. 15 (4) (1985) 580–585.
[59] H. Ahn, K. Kim, Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach, Appl. Soft Comput. 9 (2009) 599–607.
[60] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, 5th edition, Chapman & Hall/CRC, 2011.
[61] C. García-Osorio, A. de Haro-García, N. García-Pedrajas, Democratic instance selection: a linear complexity instance selection algorithm based on classifier ensemble concepts, Artif. Intell. 174 (2010) 410–441.
[62] J.R. Cano, F. Herrera, M. Lozano, Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability, Data Knowl. Eng. 60 (2007) 90–108.
[63] S. García, A. Fernández, F. Herrera, Enhancing the effectiveness and interpretability of decision tree and rule induction classifiers with evolutionary training set selection over imbalanced problems, Appl. Soft Comput. 9 (2009) 1304–1314.
[64] L. Nanni, A. Lumini, Prototype reduction techniques: a comparison among different approaches, Expert Syst. Appl. 38 (9) (2011) 11820–11828.

Isaac Triguero Velázquez received the M.Sc. degree in Computer Science from the University of Granada, Granada, Spain, in 2009. He is currently a Ph.D. student in the Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain. His research interests include data mining, semi-supervised learning, data reduction and evolutionary algorithms.

Joaquín Derrac Rus received the M.Sc. degree in Computer Science from the University of Granada, Granada, Spain, in 2008. He is currently a Ph.D. student in the Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain. His research interests include data mining, data reduction, statistical inference and evolutionary algorithms.

Salvador García López received the M.Sc. and Ph.D. degrees in Computer Science from the University of Granada, Granada, Spain, in 2004 and 2008, respectively.

He is currently an Assistant Professor in the Department of Computer Science, University of Jaén, Jaén, Spain. He has had more than 25 papers published in international journals. He has co-edited two special issues of international journals on different Data Mining topics. His research interests include data mining, data reduction, data complexity, imbalanced learning, semi-supervised learning, statistical inference and evolutionary algorithms.

Francisco Herrera Triguero received the M.Sc. in Mathematics in 1988 and the Ph.D. in Mathematics in 1991, both from the University of Granada, Spain.

He is currently a Professor in the Department of Computer Science and Artificial Intelligence at the University of Granada. He has had more than 200 papers published in international journals. He is a coauthor of the book "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases" (World Scientific, 2001). He currently acts as Editor in Chief of the international journal "Progress in Artificial Intelligence" (Springer) and serves as area editor of the Journal Soft Computing (area of evolutionary and bioinspired algorithms) and International Journal of Computational Intelligence Systems (area of information systems). He acts as associated editor of the journals: IEEE Transactions on Fuzzy Systems, Information Sciences, Advances in Fuzzy Systems, and International Journal of Applied Metaheuristics Computing; and he serves as member of several journal editorial boards, among others: Fuzzy Sets and Systems, Applied Intelligence, Knowledge and Information Systems, Information Fusion, Evolutionary Intelligence, International Journal of Hybrid Intelligent Systems, Memetic Computation, Swarm and Evolutionary Computation.

He received the following honors and awards: ECCAI Fellow 2009, 2010 Spanish National Award on Computer Science ARITMEL to the "Spanish Engineer on Computer Science", and International Cajastur "Mamdani" Prize for Soft Computing (Fourth Edition, 2010).

His current research interests include computing with words and decision making, data mining, data preparation, instance selection, fuzzy rule based systems, genetic fuzzy systems, knowledge extraction based on evolutionary algorithms, memetic algorithms and genetic algorithms.