A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification

Isaac Triguero, Joaquín Derrac, Salvador García, and Francisco Herrera

***Abstract*—The nearest neighbor (NN) rule is one of the most successfully used techniques to resolve classification and pattern recognition tasks. Despite its high classification accuracy, this rule suffers from several shortcomings in time response, noise sensitivity, and high storage requirements. These weaknesses have been tackled by many different approaches, including a good and well-known solution that we can find in the literature, which consists of the reduction of the data used for the classification rule (training data). Prototype reduction techniques can be divided into two different approaches, which are known as prototype selection and prototype generation (PG) or abstraction. The former process consists of choosing a subset of the original training data, whereas PG builds new artificial prototypes to increase the accuracy of the NN classification. In this paper, we provide a survey of PG methods specifically designed for the NN rule. From a theoretical point of view, we propose a taxonomy based on the main characteristics presented in them. Furthermore, from an empirical point of view, we conduct a wide experimental study that involves small and large datasets to measure their performance in terms of accuracy and reduction capabilities. The results are contrasted through nonparametrical statistical tests. Several remarks are made to understand which PG models are appropriate for application to different datasets.**

***Index Terms*—Classification, learning vector quantization (LVQ), nearest neighbor (NN), prototype generation (PG), taxonomy.**

I. Introduction

THE nearest neighbor (NN) algorithm [1] and its derivatives have been shown to perform well, like a nonparametric classifier, in machine-learning and data-mining (DM) tasks [2]–[4]. It is included in a more specific field of DM known as lazy learning [5], which refers to the set of methods that predicts the class label from raw training data and does not obtain learning models. Although NN is a simple technique, it has demonstrated itself to be one of the most interesting and effective algorithms in DM [6] and pattern recognition [7], and it has been considered one of the top ten methods in DM [8]. A wide range of new real problems have been stated as classifications problems [9], [10], where NN has been a great support for them, for instance, [11] and [12].

The most intuitive approach to pattern classification is based on the concept of similarity [13]–[15]; obviously, patterns that are similar, in some sense, have to be assigned to the same class. The classification process involves partitioning samples into training and testing categories. Let $\mathbf{x}_p$ be a training sample from n available samples in the training set. Let $\mathbf{x}_t$ be a test sample, ω be the true class of a training sample, and $\hat{\omega}$ be the predicted class for a test sample ($\omega, \hat{\omega} = 1, 2, \ldots, \Omega$). Here, Ω is the total number of classes. During the training process, we use only the true class ω of each training sample to train the classifier, while during testing, we predict the class $\hat{\omega}$ of each test sample. With the 1NN rule, the predicted class of test sample $\mathbf{x}_t$ is set equal to the true class ω of its NN, where $\mathbf{nn}_t$ is an NN to $\mathbf{x}_t$, if the distance

$$d(\mathbf{nn}_t, \mathbf{x}_t) = \min_i \{d(\mathbf{nn}_i, \mathbf{x}_t)\}.$$

For NN, the predicted class of test sample $\mathbf{x}_t$ is set equal to the most frequent true class among k nearest training samples. This forms the decision rule $D : \mathbf{x}_t \rightarrow \hat{\omega}$.

Despite its high classification accuracy, it is well known that NN suffers from several drawbacks [4]. Four weaknesses could be mentioned as the main causes that prevent the successful application of this classifier. The first one is the necessity of high storage requirements in order to retain the set of examples that defines the decision rule. Furthermore, the storage of all of the data instances also leads to high computational costs during the calculation of the decision rule, which is caused by multiple computations of similarities between the test and training samples. Regarding the third one, NN (especially 1NN) presents low tolerance to noise because of the fact that it considers all data relevant, even when the training set may contain incorrect data. Finally, NN makes predictions over existing data, and it assumes that input data perfectly delimits the decision boundaries among classes.

Several approaches have been suggested and studied in order to tackle the aforementioned drawbacks [16]. The research on similarity measures to improve the effectiveness of NN (and other related techniques based on similarities) is very extensive in the literature [15], [17], [18]. Other techniques reduce overlapping between classes [19] based on local probability centers, thus increasing the tolerance to noise. Researchers also investigate about distance functions that are suitable for use under high dimensionality conditions [20].

Manuscript received March 8, 2010; revised August 22, 2010 and October 25, 2010; accepted December 27, 2010. Date of publication ; date of current version. This work was supported by the Spanish Ministry of Science and Technology under Project TIN2008-06681-C06-01. The work of I. Triguero was supported by a scholarship from the University of Granada. The work of J. Derrac was supported by an FPU scholarship from the Spanish Ministry of Education and Science. This paper was recommended by Associate Editor M. Last.

I. Triguero, J. Derrac and F. Herrera are with the Department of Computer Science and Artificial Intelligence, Research Center on Information and Communications Technology, University of Granada, 18071 Granada, Spain (e-mail: triguero@decsai.ugr.es; jderrac@decsai.ugr.es; herrera@decsai.ugr.es).

S. García is with the Department of Computer Science, University of Jaén, 23071 Jaén, Spain (e-mail: sglopez@ujaen.es).

Digital Object Identifier 10.1109/TSMCC.2010.2103939

A successful technique that simultaneously tackles the computational complexity, storage requirements, and noise tolerance of NN is based on data reduction [21], [22]. These techniques aim to obtain a representative training set with a lower size compared to the original one and with a similar or even higher classification accuracy for new incoming data. In the literature, these are known as reduction techniques [21], instance selection [23]–[25], prototype selection (PS) [26], and prototype generation (PG) [22], [27], [28] (which are also known as prototype abstraction methods [29], [30]). Although the PS and PG problems are frequently confused and considered to be the same problem, each of them relate to different problems. PS methods concern the identification of an optimal subset of representative objects from the original training data by discarding noisy and redundant examples. PG methods, by contrast, besides selecting data, can generate and replace the original data with new artificial data [27]. This process allows it to fill regions in the domain of the problem, which have no representative examples in original data. Thus, PS methods assume that the best representative examples can be obtained from a subset of the original data, whereas PG methods generate new representative examples if needed, thus tackling also the fourth weakness of NN mentioned earlier.

The PG methods that we study in this survey are those specifically designed to enhance NN classification. Nevertheless, many other techniques could be used for the same goal as PG methods that are out of the scope of this survey. For instance, clustering techniques allow us to obtain a representative subset of prototypes or cluster centers, but they are obtained for more general purposes. A very good review of clustering can be found in [31].

Nowadays, there is no general categorization for PG methods. In the literature, a brief taxonomy for prototype reduction schemes was proposed in [22]. It includes both PS and PG methods and compares them in terms of classification accuracy and reduction rate. In this paper, the authors divide the prototype reduction schemes into creative (PG) and selecting methods (PS), but it is not exclusively focused on PG methods, and especially, on studying the similarities among them. Furthermore, a considerable number of PG algorithms have been proposed and some of them are rather unknown. The first approach we can find in the literature called PNN [32] is based on merging prototypes. One of the most important families of methods is that based on learning vector quantization (LVQ) [33]. Other methods are based on splitting the dimensional space [34], and even evolutionary algorithms and particle swarm optimization [35] have also been used to tackle this problem [36], [37].

Because of the absence of a focused taxonomy in the literature, we have observed that the new algorithms proposed are usually compared with only a subset of the complete family of PG methods and, in most of the studies, no rigorous analysis has been carried out.

These are the reasons that motivate the global purpose of this paper, which can be divided into three objectives.

1) To propose a new and complete taxonomy based on the main properties observed in the PG methods. The taxonomy will allow us to know the advantages and drawbacks from a theoretical point of view.
2) To make an empirical study that analyzes the PG algorithms in terms of accuracy, reduction capabilities, and time complexity. Our goal is to identify the best methods in each family, depending on the size and type of the datasets, and to stress the relevant properties of each one.
3) To illustrate through graphical representations the trend of generation performed by the schemes studied in order to justify the results obtained in the experiments.

The experimental study will include a statistical analysis based on nonparametric tests, and we will conduct experiments that involve a total of 24 PG methods, and 59 small- and large-size datasets. The graphical representations of selected data will be done by using a two-dimensional (2-D) dataset called *banana* with moderate complexity features.

This paper is organized as follows. A description of the properties and an enumeration of the methods, as well as related and advanced work on PG, are given in Section II. Section III presents the taxonomy proposed. In Section IV, we describe the experimental framework, and Section V examines the results obtained in the empirical study and presents a discussion of them. Graphical representations of generated data by PG methods are illustrated in Section VI. Finally, Section VII concludes the paper.

II. Prototype Generation: Background

PG builds new artificial examples from the training set; a formal specification of the problem is the following: Let $\mathbf{x}_p$ be an instance, where $\mathbf{x}_p = (\mathbf{x}_{p1}, \mathbf{x}_{p2}, \ldots, \mathbf{x}_{pm}, \mathbf{x}_{p\omega})$, with $\mathbf{x}_p$ belonging to a class ω given by $\mathbf{x}_{p\omega}$, and a m-dimensional space in which X_{pi} is the value of the ith feature of the pth sample. Then, let us assume that there is a training set TR, which consists of n instances $\mathbf{x}_p$, and a test set TS composed of s instances $\mathbf{x}_t$, with $\mathbf{x}_{t\omega}$ unknown. The purpose of PG is to obtain a prototype generate set TG, which consists of r, $r < n$, prototypes, which are either selected or generated from the examples of TR. The prototypes of the generated set are determined to represent efficiently 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 a NN classifier. In this paper, we will focus on the use of the NN rule, with $k = 1$, to classify the examples of TR and TS by using the TG as reference.

This section presents an overview of the PG problem. Three main topics will be discussed in the following.

1) In Section II-A, the main characteristics, which will define the categories of the taxonomy proposed in this paper, will be outlined. They refer to the type of reduction, resulting generation set, generation mechanisms, and evaluation of the search. Furthermore, some criteria to compare PG methods are established.
2) In Section II-B, we briefly enumerate all the PG methods proposed in the literature. The complete and abbreviated names will be given together with the proposed reference.
3) Finally, Section II-C explores other areas related to PG and gives an interesting summary of advanced work in this research field.

A. Main Characteristics in Prototype Generation Methods

This section establishes different properties of PG methods that will be necessary for the definition of the taxonomy in the following section. The issues discussed here include the type of reduction, resulting generation set, generation mechanisms, and evaluation of the search. Finally, some criteria will be set in order to compare the PG methods.

1) Type of Reduction: PG methods search for a reduced set TG of prototypes to represent the training set TR; there are also a variety of schemes in which the size of TG can be established.

a) *Incremental:* An incremental reduction starts with an empty reduced set TG or with only some representative prototypes from each class. Then, a succession of additions of new prototypes or modifications of earlier prototypes occurs. One important advantage of this kind of reduction is that these techniques can be faster and need less storage during the learning phase than nonincremental algorithms. Furthermore, this type of reduction allows the technique to adequately establish the number of prototypes required for each dataset. Nevertheless, this could obtain adverse results due to the requirement of a high number of prototypes to adjust TR, thus producing overfitting.

b) *Decremental:* The decremental reduction begins with $TG = TR$, and then, the algorithm starts to reduce TG or modify the prototypes in TG. It can be accomplished by following different procedures, such as merging, moving or removing prototypes, and relabeling classes. One advantage observed in decremental schemes is that all training examples are available for examination to make a decision. On the other hand, a shortcoming of these kinds of methods is that they usually present a high computational cost.

c) *Fixed:* It is common to use a fixed reduction in PG. These methods establish the final number of prototypes for TG using a user's previously defined parameter related to the percentage of retention of TR. This is the main drawback of this approach, apart from the fact that it is very dependent on each dataset tackled. However, these techniques only focus on increasing the classification accuracy.

d) *Mixed:* A mixed reduction begins with a preselected subset TG, obtained either by random selection with fixed reduction or by the run of a PS method, and then, additions, modifications, and removals of prototypes are done in TG. This type of reduction combines the advantages of the previously seen, thus allowing several rectifications to solve the problem of fixed reduction. However, these techniques are prone to overfit the data, and they usually have high computational cost.

2) Resulting Generation Set: This factor refers to the resulting set generated by the technique, i.e., whether the final set will retain border, central, or both types of points.

a) *Condensation:* This set includes the techniques, which return a reduced set of prototypes that are closer to the decision boundaries, that are also called border points. The reason behind retaining border points is that internal points do not affect the decision boundaries as much as border points and, thus, can be removed with relatively little effect on classification. The idea behind these methods is to preserve the accuracy over the training set, but the generalization accuracy over the test set can be negatively affected. Nevertheless, the reduction capability of condensation methods is normally high because of the fact that border points are less than internal points in most of the data.

b) *Edition:* These schemes instead seek to remove or modify border points. They act over points that are noisy or do not agree with their NNs, thus leaving smoother decision boundaries behind. However, such algorithms do not remove internal points that do not necessarily contribute to the decision boundaries. The effect obtained is related to the improvement of generalization accuracy in test data, although the reduction rate obtained is lower.

c) *Hybrid:* Hybrid methods try to find the smallest set TG, which maintains or even increases the generalization accuracy in test data. To achieve this, it allows modifications of internal and border points based on some specific criteria followed by the algorithm. The NN classifier is highly adaptable to these methods, obtaining great improvements, even with a very small reduced set of prototypes.

3) Generation Mechanisms: This factor describes the different mechanisms adopted in the literature to build the final TG set.

a) *Class relabeling:* This generation mechanism consists of changing the class labels of samples from TR, which could be suspicious of having errors, and belonging to other different classes. Its purpose is to cope with all types of imperfections in the training set (mislabeled, noisy, and atypical cases). The effect obtained is closely related to the improvement in generalization accuracy of the test data, although the reduction rate is kept fixed.

b) *Centroid based:* These techniques are based on generating artificial prototypes by merging a set of similar examples. The merging process is usually made from the computation of averaged attribute values over a selected set, yielding the so-called centroids. The identification and selection of the set of examples are the main concerns of the algorithms that belong to this category. These methods can obtain a high reduction rate, but they are also related to accuracy rate losses.

c) *Space splitting:* This set includes the techniques based on different heuristics to partition the feature space, along with several mechanisms to define new prototypes. The idea consists of dividing TR into some regions, which will be replaced with representative examples establishing the decision boundaries associated with the original TR. This mechanism works on a space level because of the fact that the partitions are found in order to discriminate, as well as possible, a set of examples from others, whereas centroid-based approaches work on the data level, which mainly focuses on the optimal selection of only a set of examples to be treated. The reduction capabilities of these

techniques usually depend on the number of regions that are needed to represent TR.

d) *Positioning adjustment:* The methods that belong to this family aim to correct the position of a subset of prototypes from the initial set by using an optimization procedure. New positions of prototype can be obtained by using the movement idea in the m-dimensional space, thus adding or subtracting some quantities to the attribute values of the prototypes. This mechanism is usually associated with a fixed or mixed type of reduction.

4) Evaluation of Search: The NN itself is an appropriate heuristic to guide the search of a PG method. The decisions made by the heuristic must have an evaluation measure that allows the comparison of different alternatives. The evaluation of search criterion depends on the use or nonuse of NN in such an evaluation.

a) *Filter:* We refer to filters techniques when they do not use the NN rule during the evaluation phase. Different heuristics are used to obtain the reduced set. They can be faster than NN, but the performance in terms of accuracy obtained could be worse.

b) *Semiwrapper:* NN is used for partial data to determine the criteria of making a certain decision. Thus, NN performance can be measured over localized data, which will contain most of prototypes that will be influenced in making a decision. It is an intermediate approach, where a tradeoff between efficiency and accuracy is expected.

c) *Wrapper:* In this case, the NN rule fully guides the search by using the complete training set with the leave-one-out validation scheme. The conjunction, in the use of the two mentioned factors, allows us to get a great estimator of generalization accuracy, thus obtaining better accuracy over test data. However, each decision involves a complete computation of the NN rule over the training set and the evaluation phase can be computationally expensive.

5) Criteria to Compare PG Methods: When comparing the PG methods, there are a number of criteria that can be used to compare the relative strengths and weaknesses of each algorithm. These include storage reduction, noise tolerance, generalization accuracy, and time requirements.

1) *Storage reduction:* One of the main goals of the PG methods is to reduce storage requirements. Furthermore, another goal closely related to this is to speed up classification. A reduction in the number of stored instances will typically yield a corresponding reduction in the time it takes to search through these examples and classify a new input vector.

2) *Noise tolerance:* Two main problems may occur in the presence of noise. The first is that very few instances will be removed because many instances are needed to maintain the noisy decision boundaries. Second, the generalization accuracy can suffer, especially if noisy instances are retained instead of good instances, or these are not relabeled with the correct class.

3) *Generalization accuracy:* A successful algorithm will often be able to significantly reduce the size of the training set without significantly reducing the generalization accuracy.

4) *Time requirements:* Usually, the learning process is carried out just once on a training set; therefore, it seems not to be a very important evaluation method. However, if the learning phase takes too long, it can become impractical for real applications.

TABLE I
PG METHODS REVIEWED

Complete name	Abbr. name	Reference
Prototype Nearest Neighbor	PNN	[32]
Generalized Editing using Nearest Neighbor	GENN	[39]
Learning Vector Quantization 1	LVQ1	[33]
Learning Vector Quantization 2	LVQ2	
Learning Vector Quantization 2.1	LVQ2.1	
Learning Vector Quantization 3	**LVQ3**	
Decision Surface Mapping	DSM	[40]
Vector Quantization	VQ	[41]
Chen Algorithm	Chen	[34]
Bootstrap Technique for Nearest Neighbor	BTS3	[42]
Learning Vector Quantization with Training Counter	LVQTC	[43]
MSE	MSE	[44]
Modified Chang's Algorithm	MCA	[45]
Generalized Modified Chang's Algorithm	GMCA	[46]
Integrated Concept Prototype Learner	ICPL	[29]
Integrated Concept Prototype Learner 2	**ICPL2**	
Integrated Concept Prototype Learner 2	ICPL3	
Integrated Concept Prototype Learner 2	ICPL4	
Depuration Algorithm	Depur	[47]
Hybrid LVQ3 algorithm	HYB	[48]
Reduction by space partitioning 1	RSP1	[30]
Reduction by space partitioning 2	RSP2	
Reduction by space partitioning 3	**RSP3**	
Evolutionary Nearest Prototype Classifier	ENPC	[36]
Adaptive Vector Quantization	AVQ	[49]
Learning Vector Quantization with Pruning	LVQPRU	[50]
Pairwise Opposite Class Nearest Neighbor	POC-NN	[51]
Adaptive Condensing Algorithm Based on Mixtures of Gaussians	MixtGauss	[27]
Self-generating Prototypes	SGP	[28]
Adaptive Michigan PSO	AMPSO	[52]
Prototype Selection Clonal Selection Algorithm	PSCSA	[53]
Particle Swarm Optimization	PSO	[37]

B. Prototype Generation Methods

More than 25 PG methods have been proposed in the literature. This section is devoted to enumerate and designate them according to a standard that followed in this paper. For more details on their implementations, the reader can visit the URL http://sci2s.ugr.es/pgtax. Implementations of the algorithms in java can be found in KEEL software [38].

Table I presents an enumeration of the PG methods reviewed in this paper. The complete name, abbreviation, and reference is provided for each one. In the case of there being more than one method in a row, they were proposed together and the best performing method (indicated by the respective authors) is depicted in bold. We will use the best representative method of each proposed paper; therefore, only the methods in bold, when more than one method is proposed, will be compared in the experimental study.

C. Related and Advanced Work

Nowadays, much research to enhance the NN through data preprocessing is common and highly demanded. PG could represent a feasible and promising technique to obtain expected

results, which justifies its relationship to other methods and problems. This section provides a brief review on other topics closely related to PG, and describes other interesting work and future trends, which have been studied over the past few years.

1) *Prototype selection:* With the same objective as PG, storage reduction, and classification accuracy improvement, these methods are limited only to select examples from the training set. More than 50 methods can be found in the literature. In general, three kinds of methods are usually differentiated, which are also based on edition [54], condensation [55], or hybrid models [21], [56]. Advanced proposals can be found in [24] and [57]–[59].
2) *Instance and rule learning hybridizations:* It includes all the methods, which simultaneously use instances and rules in order to compute the classification of a new object. If the values of the object are within the range of a rule, its consequent predicts the class; otherwise, if no rule matches with the object, the most similar rule or instance stored in the database is used to estimate the class. Similarity is viewed as the closest rule or instance based on a distance measure. In short, these methods can generalize an instance into a hyperrectangle or rule [60], [61].
3) *Hyperspherical prototypes:* This area [62] studies the use of hyperspheres to cover the training patterns of each class. The basic idea is to cluster the space into several objects, each of them corresponding only to one class, and the class of the nearest object is assigned to the test example.
4) *Weighting:* This task consists of applying weights to the instances of the training set, thus modifying the distance measure between them and any other instance. This technique could be integrated with the PS and PG methods [16], [63], [64], [65], [66] to improve the accuracy in classification problems and to avoid overfitting. A complete review dealing with this topic can be found in [67].
5) *Distance functions:* Several distance metrics have been used with NN, especially when working with categorical attributes [68]. Many different distance measures try to optimize the performance of NN [15], [64], [69], [70], and they have successfully increased the classification accuracy. Advanced work is based on adaptive distance functions [71].
6) *Oversampling*: This term is frequently used in learning with imbalanced classes [72], [73], and is closely related to undersampling [74]. Oversampling techniques replicate and generate artificial examples that belong to the minority classes in order to strengthen the presence of minority samples and to increase the performance over them. SMOTE [75] is the most well known oversampling technique and it has been shown to be very effective in many domains of application [76].

III. Prototype Generation: Taxonomy

The main characteristics of the PG methods have been described in Section II-A, and they can be used to categorize the PG methods proposed in the literature. The type of reduction, resulting generation set, generation mechanisms, and the evaluation of the search constitute a set of properties that define each PG method. This section presents the taxonomy of PG methods based on these properties.

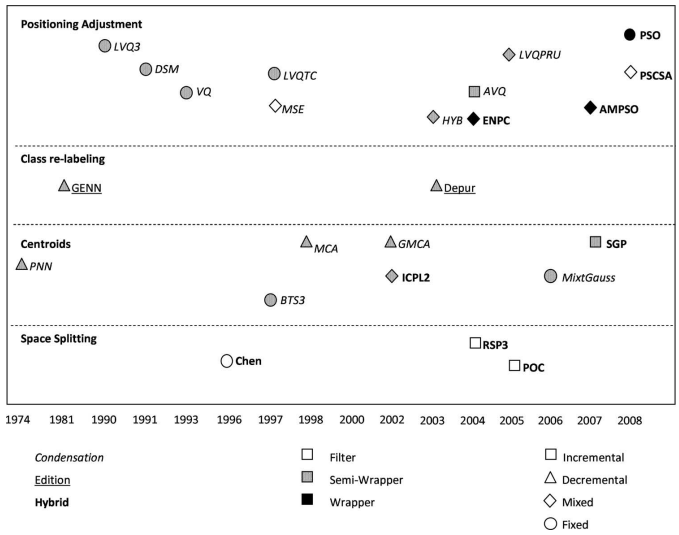


Fig. 1. Prototype generation map.

In Fig. 1, we show the PG map with the representative methods proposed in each paper ordered in time. We refer to representantive methods, which are preferred by the authors or have reported the best results in the corresponding proposal paper. Some interesting remarks can be seen in Fig. 1.

1) Only two class-relabeling methods have been proposed for PG algorithms. The reason is that both the methods obtain great results for this approach in accuracy, but the underlying concept of these methods does not achieve high reduction rates, which is one of the most important objectives of PG. Furthermore, it is important to point out that both algorithms are based on decremental reduction, and that they have noise filtering purposes.
2) The condensation techniques constitute a wide group. They usually use a semiwrapper evaluation with any type of reduction. It is considered a classic idea due to the fact that, in recent years, hybrid models are preferred over condensation techniques, with few exceptions. ICPL2 was the first PG method with a hybrid approach, combining edition, and condensation stages.
3) Recent efforts in proposing positioning adjustment algorithms are noted for mixed reduction. Most of the methods following this scheme are based on LVQ, and the recent approaches try to alleviate the main drawback of the fixed reduction.
4) There are many efforts in centroid-based techniques because they have reported a great synergy with the NN rule, since the first algorithm PNN. Furthermore, many of them are based on simple and intuitive heuristics, which allow them to obtain a reduced set with high-quality accuracy. By contrast, those with decremental and mixed reduction are slow techniques.

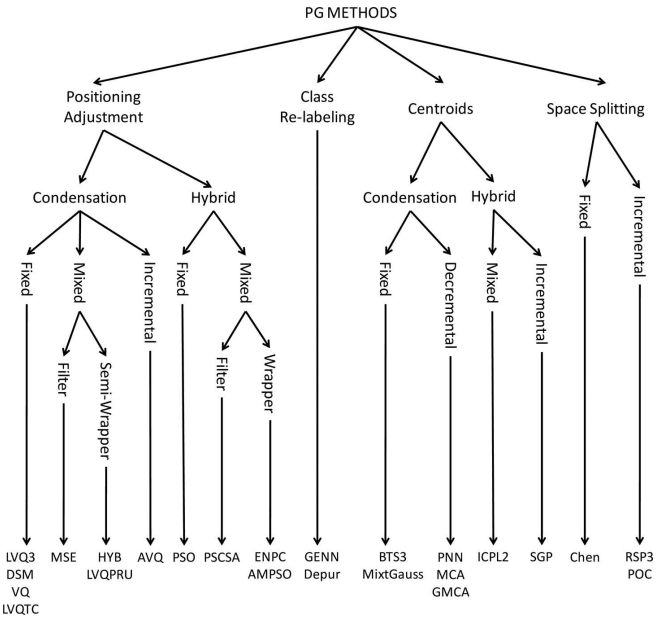


Fig. 2. Prototype generation hierarchy.

5) Wrapper evaluation appeared a few years ago and is only presented in hybrid approaches. This evaluation search is intended to optimize a selection, without taking into account computational costs.

Fig. 2 illustrates the categorization following a hierarchy based on this order: generation mechanisms, resulting generation set, type of reduction, and finally, evaluation of the search.

The properties studied here can help to understand how the PG algorithms work. In the following sections, we will establish which methods perform best, for each family, considering several metrics of performance with a wide experimental framework.

IV. Experimental Framework

In this section, we show the factors and issues related to the experimental study. We provide the measures employed to evaluate the performance of the algorithms (see Section IV-A), details of the problems chosen for the experimentation (see Section IV-B), parameters of the algorithms (see Section IV-C), and finally, the statistical tests employed to contrast the results obtained are described (see Section IV-D).

A. Performance Measures for Standard Classification

In this study, we deal with multiclass datasets. In these domains, two measures are widely used because of their simplicity and successful application. We refer to the classification rate and Cohen's kappa rate measures, which we will explain in the following.

1) *Classification rate:* It is the number of successful hits (correct classifications) relative to the total number of classifications. It has been by far the most commonly used metric to assess the performance of classifiers for years [2], [77].

2) *Cohen's Kappa (Kappa rate):* It is an alternative measure to the *classification rate*, since it compensates for random hits [78]. In contrast to the classification rate, kappa evaluates the portion of hits that can be attributed to the classifier itself (i.e., not to mere chance), relative to all the classifications that cannot be attributed to chance alone. An easy way to compute the Cohen's kappa is to makie use of the resulting confusion matrix (see Table III) in a classification task. With the following expression, we can obtain Cohen's kappa:

$$\text{kappa} = \frac{n\sum_{i=1}^{\Omega} h_{ii} - \sum_{i=1}^{\Omega} T_{ri}T_{ci}}{n^2 - \sum_{i=1}^{\Omega} T_{ri}T_{ci}} \tag{1}$$

where h_{ii} is the cell count in the main diagonal (the number of true positives for each class), n is the number of examples, Ω is the number of class labels, and T_{ri} and T_{ci} are the rows' and columns' total counts, respectively ($T_{ri} = \sum_{j=1}^{\Omega} h_{ij}, T_{ci} = \sum_{j=1}^{\Omega} h_{ji}$).

Cohen's kappa ranges from -1 (total disagreement) through 0 (random classification) to 1 (perfect agreement). For multiclass problems, kappa is a very useful, yet simple, meter to measure a classifier's classification rate while compensating for random successes.

The main difference between the classification rate and Cohen's kappa is the scoring of the correct classifications. Classification rate scores all the successes over all classes, whereas Cohen's kappa scores the successes independently for each class and aggregates them. The second way of scoring is less sensitive to randomness caused by a different number of examples in each class.

B. Datasets

In the experimental study, we selected 59 datasets from the University of California, Irvine (UCI) repository [79] and KEEL dataset[1] [38]. Table II summarizes the properties of the selected datasets. It shows, for each dataset, the number of examples (#Ex.), the number of attributes (#Atts.), the number of numerical (#Num.) and nominal (#Nom.) attributes, and the number of classes (#Cl.). The datasets are grouped into two categories depending on the size they have. Small datasets have less than 2000 instances and large datasets have more than 2000 instances. The datasets considered are partitioned by using the tenfold cross-validation (10-fcv) procedure.

C. Parameters

Many different method configurations have been established by the authors in each paper for the PG techniques. In our experimental study, we have used the parameters defined in the reference, where they were originally described, 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 IV. Note that some PG methods have no parameters to be fixed; therefore, they are not included in this table.

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

TABLE II
SUMMARY DESCRIPTION FOR CLASSIFICATION DATASETS

Dataset	#Ex.	#Atts.	#Num.	#Nom.	#Cl.	Dataset	#Ex.	#Atts.	#Num.	#Nom.	#Cl.
abalone	4,174	8	7	1	28	marketing	8,993	13	13	0	9
appendicitis	106	7	7	0	2	monks	432	6	6	0	2
australian	690	14	8	6	2	movement_libras	360	90	90	0	15
automobile	205	25	15	10	6	newthyroid	215	5	5	0	3
balance	625	4	4	0	3	nursery	12,960	8	0	8	5
banana	5,300	2	2	0	2	pageblocks	5,472	10	10	0	5
bands	539	19	19	0	2	penbased	10,992	16	16	0	10
breast	286	9	0	9	2	phoneme	5,404	5	5	0	2
bupa	345	6	6	0	2	pima	768	8	8	0	2
car	1,728	6	0	6	4	ring	7,400	20	20	0	2
chess	3,196	36	0	36	2	saheart	462	9	8	1	2
cleveland	297	13	13	0	5	satimage	6,435	36	36	0	7
coil2000	9,822	85	85	0	2	segment	2,310	19	19	0	7
contraceptive	1,473	9	9	0	3	sonar	208	60	60	0	2
crx	690	15	6	9	2	spambase	4,597	57	57	0	2
dermatology	366	33	1	32	6	spectheart	267	44	44	0	2
ecoli	336	7	7	0	8	splice	3,190	60	0	60	3
flare-solar	1,066	11	0	11	2	tae	151	5	5	0	3
german	1,000	20	7	13	2	texture	5,500	40	40	0	11
glass	214	9	9	0	7	thyroid	7,200	21	21	0	3
haberman	306	3	3	0	2	tic-tac-toe	958	9	0	9	2
hayes-roth	160	4	4	0	3	titanic	2,201	3	3	0	2
heart	270	13	13	0	2	twonorm	7,400	20	20	0	2
hepatitis	155	19	19	0	2	vehicle	846	18	18	0	4
housevotes	435	16	0	16	2	vowel	990	13	13	0	11
iris	150	4	4	0	3	wine	178	13	13	0	3
led7digit	500	7	7	0	10	wisconsin	699	9	9	0	2
lymphography	148	18	3	15	4	yeast	1484	8	8	0	10
magic	19,020	10	10	0	2	zoo	101	16	0	16	7
mammographic	961	5	0	5	2						

TABLE III
CONFUSION MATRIX FOR AN Ω-CLASS PROBLEM

Correct Class	Predicted Class				
	ω_1	ω_2	...	ω_Ω	Total
ω_1	h_{11}	h_{12}	...	$h_{1\Omega}$	T_{r1}
ω_2	h_{12}	h_{22}	...	$h_{2\Omega}$	T_{r2}
$\vdots$			$\ddots$		$\vdots$
ω_Ω	$h_{1\Omega}$	$h_{2\Omega}$	...	$h_{\Omega\Omega}$	$T_{r\Omega}$
Total	T_{c1}	T_{c2}	...	$T_{c\Omega}$	T

In most of the techniques, Euclidean distance is used as the similarity function, to decide which neighbors are closest. Furthermore, to avoid problems with a large number of attributes and distances, all datasets have been normalized between 0 and 1. This normalization process allows to apply all the PG methods over each dataset, independent of the types of attributes.

D. Statistical Tests for Performance Comparison

In this paper, we use the hypothesis-testing techniques to provide statistical support for the analysis of the results [80], [81]. Specifically, we use nonparametric tests because of the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied, thus causing the statistical analysis to lose credibility with these parametric tests. These tests are suggested in the studies presented in [80] and [82]–[84], where its use in the field of machine learning is highly recommended.

The Wilcoxon test [82], [83] is adopted considering a level of significance of $\alpha = 0.1$. More information about statistical tests and the results obtained can be found in the web site associated with this paper (http://sci2s.ugr.es/pgtax).

E. Other Considerations

We want to outline that the implementations are based only on the descriptions and specifications given by the respective authors in their papers. No advanced data structures and enhancements for improving the efficiency of PG methods have been carried out. All methods are available in KEEL software [38].

V. ANALYSIS OF RESULTS

This section presents the average results collected in the experimental study and some discussions of them; the complete

TABLE IV
PARAMETER SPECIFICATION FOR ALL THE METHODS EMPLOYED IN THE EXPERIMENTATION

Algorithm	*Parameters*
LVQ3	Iterations = 100, $alpha$ = 0.1, WindowWidth=0.2 $epsilon$ = 0.1
DSM	Iterations = 100, $alpha$ = 0.1
VQ	Iterations = 100, $alpha$ = 0.1
BTS3	NN selected = 1, Random Trials = 3
LVQTC	Iterations = 100, $alpha_R$ = 0.1, $alpha_W$ = 0.1, Retention Threshold = 3, Number of Epoches= 4
MSE	Gradient Step = 0.5, Initial Temperature = 100
ICLP2	Filtering method = RT2
Depur	$k^{'}$ = 2 , k = 3
HYB	Search Iterations = 200, Optimal search Iterations = 1000 $alpha$ = 0.1 , Initial $epsilon$ = 0, Final $epsilon$ = 0.5 Initial WindowWidth = 0, Final WindowWidth = 0.5 $delta$ = 0.1, delta WindowWidth = 0.1 Initial Selection = SVM
RSP3	Subset Choice = Diameter
ENPC	Iterations = 250
AVQ	Iterations = 100, T set percentage= 80%, $epsilon$ = 0.1
LVQPRU	Iterations = 100, $alpha$ = 0.1 ,WindowWidth = 0.5
POCNN	$alpha$ ratio = 0.2
SGP	Rmin = 0.01, Rmis = 0.2
AMPSO	Iterations = 300, C1 = 1.0, C2 = 1.0, C3 = 0.25, Vmax = 1, W = 0.1, X = 0.5, Pr = 0.1, Pd = 0.1
PSCSA	HyperMutation Rate = 2, Clonal Rate = 10, Mutation Rate = 0.01, Stimulation Threshold = 0.89, Alpha = 0.4
PSO	SwarmSize = 20, Iterations = 250, C1 = 1, C2 = 3, Vmax = 0.25, Wstart = 1.5, Wend = 0.5

Note: Parameter reduction rate on fixed reduction algorithms has been established to 95% for the small size dataset and 98% for the large

results can be found on the web page associated with this paper. The study will be divided into two parts: analysis of the results obtained over small-size datasets (see Section V-A) and over large datasets (see Section V-B). Finally, a global analysis is added in Section V-C.

A. Analysis and Empirical Results of Small-Size datasets

Table V presents the average results obtained by the PG methods over the 40 small-size datasets. *Red.* denotes reduction rate achieved, *train Acc.* and *train Kap.* present the accuracy and kappa obtained in the training data, respectively; on the other hand, *tst Acc.* and *tst Kap.* present the accuracy and kappa obtained over the test data. Finally, *Time* denotes the average time elapsed in seconds to finish a run of PG method. The algorithms are ordered from the best to the worst for each type of result. Algorithms highlighted in bold are those which obtain the best result in their corresponding family, according to the first level of the hierarchy in Fig. 2.

Fig. 3 depicts a representation of an opposition between the two objectives: reduction and test accuracy. Each algorithm located inside the graphic gets its position from the average values of each measure evaluated (exact position corresponding to the beginning of the name of the algorithm). Across the graphic, there is a line that represents the threshold of test accuracy achieved by the 1NN algorithm without preprocessing. Note that in Fig. 3(a), the names of some PG methods overlap, and hence, Fig. 3(b) shows this overlapping zone.

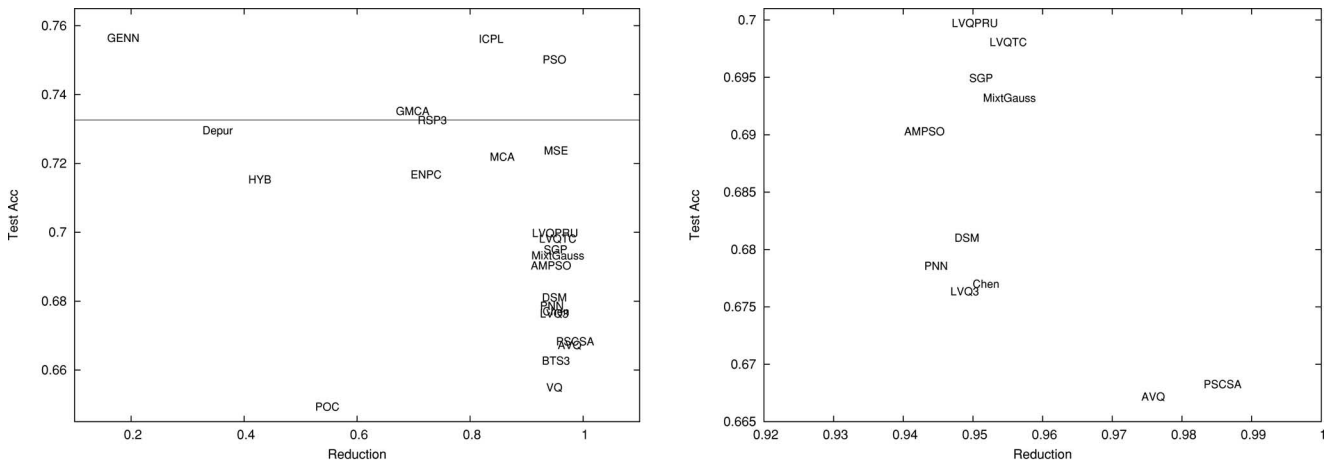
To complete the set of results, the web site associated with this paper contains the results of applying the Wilcoxon test to all possible comparisons among all PG considered in small datasets.

Observing Table V, Fig. 3, and the Wilcoxon Test, we can point out some interesting facts as follows.

1) Some classical algorithms are at the top in accuracy and kappa rate. For instance, GENN, GMCA, and MSE obtain better results than other recent methods over test data. However, these techniques usually have a poor associated reduction rate. We can observe this statement in the Wilcoxon test, where classical methods significantly overcome other recent approaches in terms of accuracy and kappa rates. However, In terms of $Acc. * Red.$ and $Kap. * Red.$ measures, typically, these methods do not outperform recent techniques.
2) PSO and ENPC could be stressed from the *positioning adjustment* family as the best performing methods. Each one of them belongs to different subfamilies, fixed and mixed reduction, respectively. PSO focuses on improving the classification accuracy, and it obtains a good generalization capability. On the other hand, ENPC has the overfitting as the main drawback, which is clearly discernible from Table V. In general, LVQ-based approaches obtain worse accuracy rates than 1NN, but the reduction rate achieved by them is very high. MSE and HYB are the most outstanding techniques belonging to the subgroup of *condensation* and *positioning adjustment*.
3) With respect to *class-relabeling* methods, GENN obtains better accuracy/kappa rates but worse reduction rates than Depur. However, the statistical test informs that GENN does not outperform to the Depur algorithm in terms of accuracy and kappa rate. Furthermore, when the reduction rate is taken into consideration, i.e., when the statistical test is based on the $Acc. * Red.$ and $Kap. * Red.$ measures, the Depur algorithm clearly outperforms GENN.
4) The decremental approaches belonging to the *centroids* family require high computation times, but usually offer good reduction rates. MCA and PNN tend to overfit the data, but GMCA obtains excellent results.
5) In the whole *centroids* family, two methods deserve particular mention: ICPL2 and GMCA. Both generate a reduced prototype set with good accuracy rates in test data. The other approaches based on fixed and incremental reduction are less appropriate to improve the effectiveness of 1NN, but they are very fast and offer much reduced generated sets.
6) Regarding *space-splitting* approaches, several differences can be observed. RSP3 is an algorithm based on Chen's algorithm, but tries to avoid drastic changes in the form of the decision boundaries, and it produces a good tradeoff between reduction and accuracy. Although the POC algorithm is a relatively modern technique, this does not obtain great results. We can justify these results because the α-parameter is very sensitive for each dataset. Furthermore,

TABLE V
AVERAGE RESULTS OBTAINED BY THE PG METHODS OVER SMALL DATASETS

Red.		**train Acc.**		**train Kap.**		**tst Acc.**		**tst Kap.**		**Time (s)**	
PSCSA	0.9858	**MCA**	0.8772	**MCA**	0.7717	**GENN**	0.7564	**GENN**	0.5400	1NN	-
AVQ	0.9759	GMCA	0.8405	GMCA	0.7067	**ICPL2**	0.7560	**ICPL2**	0.5366	LVQTC	0.1644
LVQTC	0.9551	HYB	0.8309	HYB	0.6988	**PSO**	0.7501	**PSO**	0.5332	DSM	0.1780
MixtGauss	0.9552	ICPL2	0.8254	**ENPC**	0.6800	GMCA	0.7351	GMCA	0.5062	BTS3	0.2079
MSE	0.9520	**ENPC**	0.8247	PSO	0.6791	1NN	0.7326	**RSP3**	0.5004	LVQ3	0.2316
Chen	0.9519	PSO	0.8238	ICPL2	0.6690	**RSP3**	0.7325	MSE	0.4925	VQ	0.2469
BTS3	0.9519	**GENN**	0.8002	**GENN**	0.6243	Depur	0.7296	1NN	0.4918	Chen	0.2675
SGP	0.9512	**RSP3**	0.7924	**RSP3**	0.6112	MSE	0.7237	MCA	0.4867	Depur	0.2777
LVQPRU	0.9503	Depur	0.7801	Depur	0.5815	MCA	0.7219	Depur	0.4826	LVQPRU	0.5592
PSO	0.9491	MSE	0.7566	MSE	0.5388	ENPC	0.7167	ENPC	0.4818	AVQ	0.6561
VQ	0.9491	1NN	0.7369	LVQTC	0.5224	HYB	0.7153	HYB	0.4790	MixtGauss	0.8125
DSM	0.9491	LVQTC	0.7327	Chen	0.5116	LVQPRU	0.6997	LVQPRU	0.4592	SGP	1.3597
LVQ3	0.9488	LVQPRU	0.7304	LVQPRU	0.5110	LVQTC	0.6981	MixtGauss	0.4546	GENN	1.4285
PNN	0.9447	SGP	0.7256	AMPSO	0.5039	SGP	0.6949	LVQTC	0.4541	RSP3	1.8505
AMPSO	0.9430	AMPSO	0.7227	1NN	0.4985	MixtGauss	0.6932	AMPSO	0.4440	PSCSA	1.9562
MCA	0.8568	MixtGauss	0.7138	MixtGauss	0.4888	AMPSO	0.6903	PNN	0.4369	MSE	2.4794
ICPL2	0.8371	DSM	0.7036	SGP	0.4852	DSM	0.6810	SGP	0.4360	HYB	5.5888
RSP3	0.7329	PNN	0.7015	PNN	0.4718	PNN	0.6786	AVQ	0.4326	AMPSO	8.2870
ENPC	0.7220	Chen	0.6964	AVQ	0.4660	Chen	0.6770	DSM	0.4239	GMCA	8.4947
GMCA	0.6984	LVQ3	0.6931	DSM	0.4627	LVQ3	0.6763	PSCSA	0.4231	PNN	14.0066
POC	0.6071	AVQ	0.6869	PSCSA	0.4461	PSCSA	0.6682	LVQ3	0.4114	PSO	42.3168
HYB	0.4278	PSCSA	0.6787	LVQ3	0.4421	AVQ	0.6672	Chen	0.4026	ENPC	47.1377
Depur	0.3531	BTS3	0.6713	BTS3	0.3923	BTS3	0.6626	BTS3	0.3784	POC	151.9278
GENN	0.1862	VQ	0.6614	VQ	0.3866	VQ	0.6549	VQ	0.3770	ICPL2	163.9147
1NN	0.0000	POC	0.6487	POC	0.3601	POC	0.6493	POC	0.3700	MCA	190.4930



(a) (b)

Fig. 3. Accuracy in test versus reduction in small datasets. (a) All PG methods. (b) Zoom in the overlapping reduction-rate zone.

it is quite slow when tackling datasets with more than two classes.

7) The best methods in accuracy/kappa rates for each one of the families are PSO, GENN, ICPL2, and RSP3, respectively, and five methods outperform 1NN in accuracy.
8) In general, hybrid methods obtain the best result in terms of accuracy and reduction rate.
9) Usually, there is no difference between the rankings obtained with accuracy and kappa rates, except for some concrete algorithms. For example, we can observe that 1NN obtains a lower ranking with the kappa measure; it probably indicates that 1NN benefits from random hits.

Furthermore, in the web site associated with this paper, we can find an analysis of the results depending on the type of attributes of the datasets. We show the results in accuracy/kappa rate for all PG methods differentiating between numerical, nominal, and mixed datasets. In numerical and nominal datasets, all attributes must be numerical and nominal, respectively, whereas in mixed datasets, we include those datasets with numerical and nominal attributes mixed. Observing these tables, we want to outline different properties of the PG methods.

1) In general, there is no difference in performance between numerical, nominal, and mixed datasets, except for some concrete algorithms. For example, in mixed datasets, we

TABLE VI
AVERAGE RESULTS OBTAINED BY THE PG METHODS OVER LARGE DATASETS

Red.		**train Acc.**		**train Kap.**		**tst Acc.**		**tst Kap.**		**Time (s)**	
PSCSA	0.9988	**ENPC**	0.8809	**ENPC**	0.7613	**GENN**	0.8133	**GENN**	0.6269	1NN	
AVQ	0.9980	**GENN**	0.8428	**GENN**	0.6806	1NN	0.8060	1NN	0.6181	DSM	1.6849
LVQTC	0.9975	Depur	0.8250	Depur	0.6322	**ENPC**	0.8029	**ENPC**	0.6170	LVQ3	1.7037
MSE	0.9936	PSO	0.8158	**RSP3**	0.6299	Depur	0.8004	Depur	0.5863	VQ	1.7193
SGP	0.9823	1NN	0.8057	1NN	0.6178	PSO	0.8000	PSO	0.5861	MSE	17.4228
BTS3	0.9801	**RSP3**	0.7922	PSO	0.6173	MSE	0.7674	**RSP3**	0.5597	HYB	18.6338
Mixtgauss	0.9801	HYB	0.7888	HYB	0.5992	Chen	0.7621	HYB	0.5567	LVQPRU	24.4067
LVQPRU	0.9801	MSE	0.7759	MSE	0.5349	HYB	0.7618	MSE	0.5221	Depur	26.8656
Chen	0.9801	Chen	0.7682	Chen	0.5236	**RSP3**	0.7556	Chen	0.5116	AVQ	38.3665
LVQ3	0.9799	AMPSO	0.7436	**BTS3**	0.4859	AMPSO	0.7410	LVQPRU	0.4799	Chen	50.0435
DSM	0.9799	**BTS3**	0.7393	AMPSO	0.4836	**BTS3**	0.7399	DSM	0.4796	SGP	52.3400
VQ	0.9799	LVQPRU	0.7373	LVQPRU	0.4818	LVQPRU	0.7356	**BTS3**	0.4788	LVQTC	83.6030
PSO	0.9799	DSM	0.7353	DSM	0.4795	DSM	0.7341	AMPSO	0.4784	PSCSA	160.3864
AMPSO	0.9797	MixtGauss	0.7345	Mixtgauss	0.4711	Mixtgauss	0.7318	MixtGauss	0.4661	GENN	167.4849
ENPC	0.8205	LVQ3	0.7340	VQ	0.4689	LVQ3	0.7318	VQ	0.4651	BTS3	219.2394
RSP3	0.8100	VQ	0.7322	LVQ3	0.4683	VQ	0.7316	LVQ3	0.4627	AMPSO	587.7181
HYB	0.5727	LVQTC	0.7065	AVQ	0.4321	LVQTC	0.7056	AVQ	0.4280	RSP3	258.6881
Depur	0.2708	PSCSA	0.6730	LVQTC	0.4185	PSCSA	0.6707	LVQTC	0.4165	MixtGauss	639.3139
GENN	0.1576	AVQ	0.6546	PSCSA	0.3900	AVQ	0.6518	PSCSA	0.3842	PSO	909.9820
1NN	0.0000	SGP	0.6162	SGP	0.3568	SGP	0.6086	SGP	0.3466	ENPC	10931.1977

can see that a class-relabeling method, GENN, is on the top because of the fact that it does not produce modifications to the attributes. However, in numerical datasets, PSO is the best performing method, indicating to us that the positioning adjustment strategy is usually well adapted to numerical datasets.

2) In fact, comparing these tables, we observe that some representative techniques of the positioning adjustment family, such as PSO, MSE, and ENPC, have an accuracy/kappa rate close to 1NN. However, over nominal and mixed datasets, they decrease their accuracy rates.
3) ICPL2 and GMCA techniques obtain good accuracy/kappa rates independent of the type of input data.

Finally, we perform a study depending on the number of classes of the datasets. In the web site associated with this paper, we show the average results in accuracy/kappa rate differentiating between binary and multiclass datasets. We can analyze several details from the results collected, which are as follows.

1) Eight techniques outperform 1NN in accuracy when they tackle binary datasets. However, over multiclass datasets, there are only three techniques that are able to overcome 1NN.
2) Centroid-based techniques usually perform well when dealing with multiclass datasets. For instance, we can highlight the MCA, SGP, PNN, ICPL2, and GMCA techniques, which increase their respective rankings with multiclass datasets.
3) GENN and ICPL2 techniques obtain good accuracy/kappa rates independent of the number of classes.
4) PSCSA has a good behavior with binary datasets. However, over multiclass datasets, PSCSA decreases its performance.
5) Some methods present significant differences between accuracy and kappa measures when dealing with binary datasets. We can stress MSE, Depur, Chen, and BTS3 like techniques penalized by the kappa measure.

B. Analysis and Empirical Results of Large-Size Datasets

This section presents the study and analysis of large-size datasets. The goal of this study is to analyze the effect of scaling up the data in PG methods. For time complexity reasons, several algorithms cannot be run over large datasets. PNN, MCA, GMCA, ICPL2, and POC are extremely slow techniques, and their time complexity quickly increases when the data scale up or manage more than five classes.

Table VI shows the average results obtained, and Fig. 4 illustrates the comparison between the accuracy and reduction rates of the PG methods over large-size datasets. Finally, the web site associated with this paper contains the results of applying the Wilcoxon test over all possible comparisons among all PG considered in large datasets.

These tables allow us to highlight some observations of the results obtained as follows.

1) Only the GENN approach outperforms the performance of the 1NN in accuracy/kappa rate.
2) Some methods present clear differences when dealing with large datasets. For instance, we can highlight the PSO and RSP3 techniques. The former may suffer from a lack of convergence due to the fact that the performance obtained in training data is slightly higher than that obtained by 1NN; hence, it may be a sign that more iterations are needed to tackle large datasets. On the other hand, the techniques based on space partitioning present some drawbacks when the data scale up and are made up of more attributes. This is the case with RSP3.
3) In general, LVQ-based methods do not work well when the data scale up.

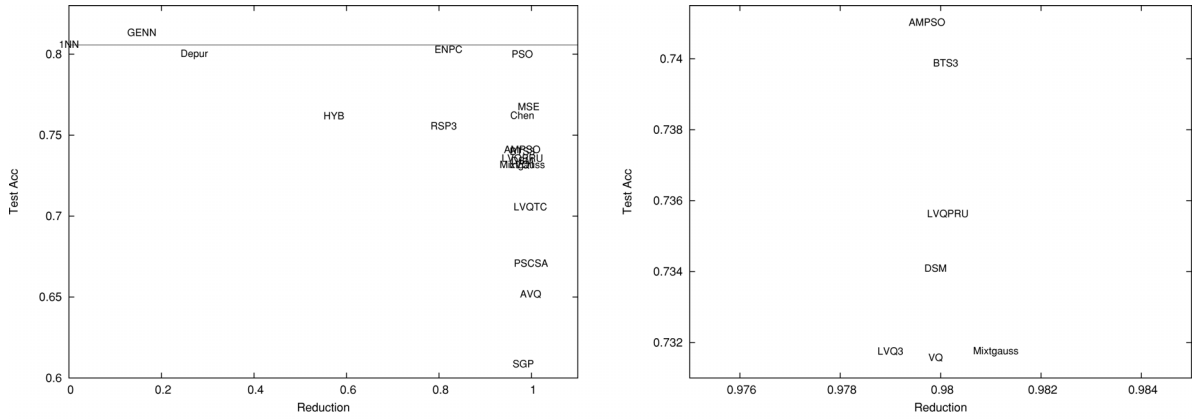


Fig. 4. Accuracy in test versus reduction in large datasets. (a) All PG methods considered over large datasets. (b) Zoom in the overlapping reduction-rate zone.

4) BTS3 stands out as the best centroids-based method over large-size datasets because the best performing ones over small datasets were also the most complex in time, and they cannot be run here.
5) Although ENPC overfits the data, it is the best performing method that consider the tradeoff between accuracy/kappa and reduction rates. PSO can also be stressed as a good candidate in this type of dataset.
6) There is no significant differences between the accuracy and kappa rankings when dealing with large datasets.

Again, we differentiate between numerical, nominal, and mixed datasets. Complete results can be found in the web site associated with this paper. Observing these results, we want to outline different properties of PG methods over large datasets. Note that there is only one dataset with mixed attributes; for this reason, we focus this analysis on the differences between numerical and nominal datasets.

1) When only numerical datasets are taken into consideration, three algorithms outperform the 1NN rule: GENN, PSO, and ENPC.
2) Over nominal large datasets, no PG method outperforms 1NN.
3) MixtGauss and AMPSO are highly conditioned on the type of input data, preferring numerical datasets. By contrast, RSP3 is better adapted to nominal datasets.

Finally, we perform again an analysis of the behavior of the PG techniques depending on the number of classes, but in this case, over large datasets. the web site associated with this paper presents the results. Observing these results, we can point out several comments.

1) Over binary large datasets, there are four algorithms that outperform 1NN. However, when the PG techniques tackle multiclass datasets, no PG method overcome 1NN.
2) When dealing with large datasets, there is no important differences between the accuracy and kappa ranking with binary datasets.
3) Class-relabeling methods perform well independent of the number of classes.

C. Global Analysis

This section shows a global view of the obtained results. As a summary, we want to outline several remarks on the use of PG because the choice of a certain method depends on various factors.

1) Several PG methods can be emphasized according to their test accuracy/kappa obtained: PSO, ICPL2, ENPC, and GENN. In principle, in terms of reduction capabilities, PSCSA and AVQ obtain the best results, but they offer poor accuracy rates. Taking into consideration the computational cost, we can consider DSM, LVQ3, and VQ to be the fastest algorithms.
2) Edition schemes usually outperform the 1NN classifier, but the number of prototypes in the result set is too high. This fact could be prohibitive over large datasets because there is no significant reduction. Furthermore, other PG methods have shown that it is possible to preserve high accuracy with a better reduction rate.
3) A high reduction rate serves no purpose, if there is no minimum guarantee of performance accuracy. This is the case of PSCSA or AVQ. Nevertheless, MSE offers excellent reduction rates without losing performance accuracy.
4) For the tradeoff reduction–accuracy rate, PSO has been reported to have the best results over small-size datasets. In the case of dealing with large datasets, the ENPC approach seems to be the most appropriate one.
5) A good reduction–accuracy balance is difficult to achieve with a fast algorithm. Considering this restriction, we could say that RSP3 allows us to yield generated sets with a good tradeoff among reduction, accuracy, and time complexity.

VI. Visualization of Data Resulting Sets: A Case Study Based on Banana Dataset

This section is devoted to illustrate the subsets selected resulting from some PG algorithms considered in this study. To do this, we focus on the *banana* dataset, which contains 5300 examples in the complete set. It is an artificial dataset of two

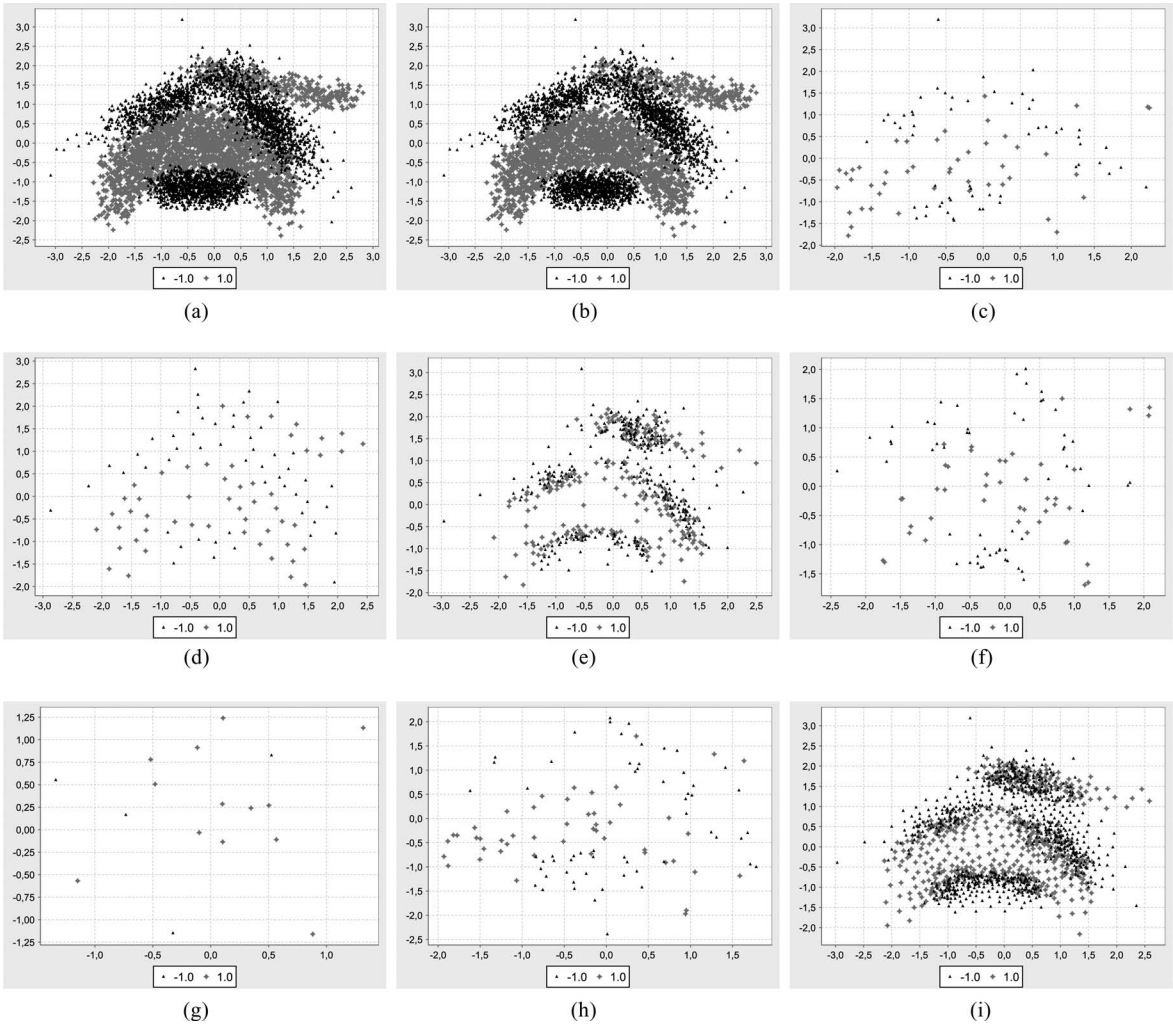


Fig. 5. Data generated sets in banana dataset. (a) Banana original $(0.8751, 0.7476)$. (b) GENN $(0.0835, 0.8826, 0.7626)$. (c) LVQ3 $(0.9801, 0.8370, 0.6685)$. (d) Chen $(0.9801, 0.8792, 0.7552)$. (e) RSP3 $(0.8962, 0.8755, 0.7482)$. (f) BTS3 $(0.9801, 0.8557, 0.7074)$. (g) SGP $(0.9961, 0.6587, 0.3433)$. (h) PSO $(0.9801, 0.8819, 0.7604)$. (i) ENPC $(0.7485, 0.8557, 0.7086)$.

classes composed of three well-defined clusters of instances of the class -1 and two clusters of the class 1. Although the borders are clear among the clusters, there is a high overlap between both classes. The complete dataset is illustrated in Fig. 5(a).

The pictures of the generated sets by some PG methods could help to visualize and understand their way of working and the results obtained in the experimental study. The reduction rate and the accuracy and kappa values in test data registered in the experimental study are specified for each one. In the original dataset, the two values indicated correspond to accuracy and kappa with 1NN.

1) Fig. 5(b) depicts the generated data by the algorithm GENN. It belongs to the edition approaches, and the generated subset differs slightly from the original dataset. Those samples found within the class boundaries can either be removed or be relabeled. It is noticeable that the clusters of different classes are a little more separated.
2) Fig. 5(c) shows the resulting subset of the classical LVQ3 condensation algorithm. It can be appreciated that most of the points are moved to define the class boundaries, but a few interior points are also used. The accuracy and kappa decrease with respect to the original, as is usually the case with condensation algorithms.
3) Fig. 5(d) and (e) represents the sets generated by the Chen and RSP3 methods, respectively. These methods are based on a space-splitting strategy, but the first one requires the specification of the final size of the generated sets, while the latter does not. We can see that the Chen method generates prototypes keeping a homogeneous distribution of points in the space. RSP3 was proposed to fix some problems observed in the Chen method, but in this concrete dataset, this method is worse in accuracy/kappa rates than its ancestor. However, the reduction type of Chen's method is fixed, and it is very dependent on the dataset tackled.
4) Fig. 5(f) and (g) represents the sets of data generated by BTS3 and SGP methods. Both techniques are cluster-based and present very high reduction rates over this dataset. SGP does not work well in this dataset because it promotes the removal of prototypes and uses an incremental order, which does not allow us to choose the most

appropriate decision. BTS3 uses a fixed reduction type; thus, it focuses on improving accuracy rates, but its generation mechanisms are not well suited for this type of dataset.

5) Fig. 5(h) and (i) illustrates the sets of data generated by PSO and ENPC methods. They are wrapper and hybrid methods of the position-adjusting family and iterate many times to obtain an optimal reallocation of prototypes. PSO requires the final size of the subset selected as a parameter, and this parameter is very conditioned to the complexity of the dataset addressed. In the *banana* case, keeping 2% of prototypes seems to work well. On the other hand, ENPC can adjust the number of prototypes required to fit a specific dataset. In the case study presented, we can see that it obtains similar sets to those obtained by the Chen approach because it also fills the regions with a homogeneous distribution of generated prototypes. In decision boundaries, the density of prototypes is increased and may produce quite noisy samples for further classification of the test data. It explains its poor behavior in this problem with respect to PSO, the lower reduction rate achieved, and the decrement of accuracy/kappa rates with regard to the original dataset classified with 1NN.

We have seen the resulting datasets of condensation, edition, and hybrid methods and different generation mechanisms with some representative PG methods. Although the methods can be categorized as a specific family, they do not follow a specific behavior pattern, since some of the condensation techniques may generate interior points (like in LVQ3), other clusters of data (RSP3), or even points with a homogeneous distribution in space (Chen or ENPC). Nevertheless, visual characteristics of generated sets are also the subject of interest and can also help to decide the choice of a PG method.

VII. Conclusion

In this paper, we have provided an overview of the PG methods proposed in the literature. We have identified the basic and advanced characteristics. Furthermore, existing work and related fields have been reviewed. Based on the main characteristics studied, we have proposed a taxonomy of the PG methods.

The most important methods have been empirically analyzed over small and large sizes of classification datasets. To illustrate and strengthen the study, some graphical representations of data subsets selected have been drawn and statistical analysis based on nonparametric tests has been employed. Several remarks and guidelines can be suggested.

1) A researcher who needs to apply a PG method should know the main characteristics of these kinds of methods in order to choose the most suitable. The taxonomy proposed and the empirical study can help a researcher to make this decision.
2) To propose a new PG method, rigorous analysis should be considered to compare the most well-known approaches and those which fit with the basic properties of the new proposal. To do this, the taxonomy and analysis of influence in the literature can help guide a future proposal to the correct method.
3) This paper helps nonexperts in PG methods to differentiate between them, to make an appropriate decision about their application, and to understand their behavior.
4) It is important to know the main advantages of each PG method. In this paper, many PG methods have been empirically analyzed, but a specific conclusion cannot be drawn regarding the best performing method. This choice depends on the problem tackled, but the results offered in this paper could help to reduce the set of candidates.

References

[1] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, Jan. 1967.
[2] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. Cambridge, MA: MIT Press, 2010.
[3] V. Cherkassky and F. Mulier, *Learning From Data: Concepts, Theory and Methods*, 2nd ed. New York: Interscience, 2007.
[4] I. Kononenko and M. Kukar, *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. West Sussex: Horwood, 2007.
[5] E. K. Garcia, S. Feldman, M. R. Gupta, and S. Srivastava, "Completely lazy learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 9, pp. 1274–1285, Sep. 2010.
[6] A. N. Papadopoulos and Y. Manolopoulos, *Nearest Neighbor Search: A Database Perspective*. New York: Springer-Verlag, 2004.
[7] G. Shakhnarovich, T. Darrell, and P. Indyk, Eds., *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. Cambridge, MA: MIT Press, 2006.
[8] X. Wu and V. Kumar, Eds., *The Top Ten Algorithms in Data Mining*. (Chapman & Hall/CRC Data Mining and Knowledge Discovery Series). Boca Raton, FL: CRC, 2009.
[9] A. Shintemirov, W. Tang, and Q. Wu, "Power transformer fault classification based on dissolved gas analysis by implementing bootstrap and genetic programming," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 39, no. 1, pp. 69–79, Jan. 2009.
[10] P. G. Espejo, S. Ventura, and F. Herrera, "A survey on the application of genetic programming to classification," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 2, pp. 121–144, Mar. 2009.
[11] S. Magnussen, R. McRoberts, and E. Tomppo, "Model-based mean square error estimators for k-nearest neighbour predictions and applications using remotely sensed data for forest inventories," *Remote Sens. Environ.*, vol. 113, no. 3, pp. 476–488, 2009.
[12] M. Govindarajan and R. Chandrasekaran, "Evaluation of k-nearest neighbor classifier performance for direct marketing," *Expert Syst. Appl.*, vol. 37, no. 1, pp. 253–258, 2009.
[13] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.
[14] Y. Chen, E. Garcia, M. Gupta, A. Rahimi, and L. Cazzanti, "Similarity-based classification: Concepts and algorithms," *J. Mach. Learning Res.*, vol. 10, pp. 747–776, 2009.
[15] K. Weinberger and L. Saul, "Distance metric learning for large margin nearest neighbor lassification," *J. Mach. Learning Res.*, vol. 10, pp. 207–244, 2009.
[16] F. Fernández and P. Isasi, "Local feature weighting in nearest prototype classification," *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 40–53, Jan. 2008.
[17] E. Pekalska and R. P. Duin, "Beyond traditional kernels: Classification in two dissimilarity-based representation spaces," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 6, pp. 729–744, Nov. 2008.
[18] P. Cunningham, "A taxonomy of similarity mechanisms for case-based reasoning," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 11, pp. 1532–1543, Nov. 2009.
[19] B. Li, Y. W. Chen, and Y. Chen, "The nearest neighbor algorithm of local probability centers," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 38, no. 1, pp. 141–154, Feb. 2008.
[20] C.-M. Hsu and M.-S. Chen, "On the design and applicability of distance functions in high-dimensional data space," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 4, pp. 523–536, Apr. 2009.

[21] D. R. Wilson and T. R. Martinez, "Reduction techniques for instance-based learning algorithms," *Mach. Learning*, vol. 38, no. 3, pp. 257–286, 2000.

[22] S. W. Kim and J. Oomenn, "A brief taxonomy and ranking of creative prototype reduction schemes," *Pattern Anal. Appl.*, vol. 6, pp. 232–244, 2003.

[23] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," *Data Mining Knowl. Discov.*, vol. 6, no. 2, pp. 153–172, 2002.

[24] E. Marchiori, "Class conditional nearest neighbor for large margin instance selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 364–370, Feb. 2010.

[25] N. García-Pedrajas, "Constructing ensembles of classifiers by means of weighted instance selection," *IEEE Trans. Neural Netw.*, vol. 20, no. 2, pp. 258–277, Feb. 2009.

[26] E. Pekalska, R. P. W. Duin, and P. Paclík, "Prototype selection for dissimilarity-based classifiers," *Pattern Recognit.*, vol. 39, no. 2, pp. 189–208, 2006.

[27] M. Lozano, J. M. Sotoca, J. S. Sánchez, F. Pla, E. Pekalska, and R. P. W. Duin, "Experimental study on prototype optimisation algorithms for prototype-based classification in vector spaces," *Pattern Recognit.*, vol. 39, no. 10, pp. 1827–1838, 2006.

[28] H. A. Fayed, S. R. Hashem, and A. F. Atiya, "Self-generating prototypes for pattern classification," *Pattern Recognit.*, vol. 40, no. 5, pp. 1498–1509, 2007.

[29] W. Lam, C. K. Keung, and D. Liu, "Discovering useful concept prototypes for classification based on filtering and abstraction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 8, pp. 1075–1090, Aug. 2002.

[30] J. S. Sánchez, "High training set size reduction by space partitioning and prototype abstraction," *Pattern Recognit.*, vol. 37, no. 7, pp. 1561–1564, 2004.

[31] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[32] C.-L. Chang, "Finding prototypes for nearest neighbor classifiers," *IEEE Trans. Comput.*, vol. C-23, no. 11, pp. 1179–1184, Nov. 1974.

[33] T. Kohonen, "The self organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.

[34] C. H. Chen and A. Jóźwik, "A sample set condensation algorithm for the class sensitive artificial neural network," *Pattern Recognit. Lett.*, vol. 17, no. 8, pp. 819–823, Jul. 1996.

[35] R. Kulkarni and G. Venayagamoorthy, "Particle swarm optimization in wireless-sensor networks: A brief survey ," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.* to be published. DOI: 10.1109/TSMCC. 2010.2054080.

[36] F. Fernández and P. Isasi, "Evolutionary design of nearest prototype classifiers," *J. Heurist.*, vol. 10, no. 4, pp. 431–454, 2004.

[37] L. Nanni and A. Lumini, "Particle swarm optimization for prototype reduction," *Neurocomputing*, vol. 72, no. 4–6, pp. 1092–1097, 2008.

[38] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, and F. Herrera, "KEEL: A software tool to assess evolutionary algorithms for data mining problems," *Soft Comput.*, vol. 13, no. 3, pp. 307–318, 2008.

[39] J. Koplowitz and T. Brown, "On the relation of performance to editing in nearest neighbor rules," *Pattern Recognit.*, vol. 13, pp. 251–255, 1981.

[40] S. Geva and J. Sitte, "Adaptive nearest neighbor pattern classification," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 318–322, Mar. 1991.

[41] Q. Xie, C. A. Laszlo, and R. K. Ward, "Vector quantization technique for nonparametric classifier design," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 12, pp. 1326–1330, Dec. 1993.

[42] Y. Hamamoto, S. Uchimura, and S. Tomita, "A bootstrap technique for nearest neighbor classifier design," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 1, pp. 73–79, Jan. 1997.

[43] R. Odorico, "Learning vector quantization with training count (LVQTC)," *Neural Netw.*, vol. 10, no. 6, pp. 1083–1088, 1997.

[44] C. Decaestecker, "Finding prototypes for neares neghbour classification by means of gradient descent and deterministic annealing," *Pattern Recognit.*, vol. 30, no. 2, pp. 281–288, 1997.

[45] T. Bezdek, J.C .and Reichherzer, G. Lim, and Y. Attikiouzel, "Multiple prototype classifier design," *IEEE Trans. Syst., Man Cybern. C, Appl. Rev.*, vol. 28, no. 1, pp. 67–79, Feb. 1998.

[46] R. Mollineda, F. Ferri, and E. Vidal, "A merge-based condensing strategy for multiple prototype classifiers," *IEEE Trans. Syst., Man Cybern. B, Cybern.*, vol. 32, no. 5, pp. 662–668, Oct. 2002.

[47] J. S. Sánchez, R. Barandela, A. I. Marqués, R. Alejo, and J. Badenas, "Analysis of new techniques to obtain quality training sets," *Pattern Recognit. Lett.*, vol. 24, no. 7, pp. 1015–1022, 2003.

[48] S. W. Kim and J. Oomenn, "Enhancing prototype reduction schemes with lvq3-type algorithms," *Pattern Recognit.*, vol. 36, pp. 1083–1093, 2003.

[49] C.-W. Yen, C.-N. Young, and M. L. Nagurka, "A vector quantization method for nearest neighbor classifier design," *Pattern Recognit. Lett.*, vol. 25, no. 6, pp. 725–731, 2004.

[50] J. Li, M. T. Manry, C. Yu, and D. R. Wilson, "Prototype classifier design with pruning," *Int. J. Artif. Intell. Tools*, vol. 14, no. 1–2, pp. 261–280, 2005.

[51] T. Raicharoen and C. Lursinsap, "A divide-and-conquer approach to the pairwise opposite class-nearest neighbor (POC-NN) algorithm," *Pattern Recognit. Lett.*, vol. 26, no. 10, pp. 1554–1567, 2005.

[52] A. Cervantes, I. M. Galván, and P. Isasi, "AMPSO: A new particle swarm method for nearest neighborhood classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 5, pp. 1082–1091, Oct. 2009.

[53] U. Garain, "Prototype reduction using an artificial immune model," *Pattern Anal. Appl.*, vol. 11, no. 3–4, pp. 353–363, 2008.

[54] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Trans. Syst., Man Cybern.*, vol. SMC-2, no. 3, pp. 408–421, Jul. 1972.

[55] P. E. Hart, "The condensed nearest neighbor rule," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 3, pp. 515–516, May 1968.

[56] S. García, J.-R. Cano, E. Bernadó-Mansilla, and F. Herrera, "Diagnose of effective evolutionary prototype selection using an overlapping measure," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 28, no. 8, pp. 1527–1548, 2009.

[57] S. García, J. R. Cano, and F. Herrera, "A memetic algorithm for evolutionary prototype selection: A scaling up approach," *Pattern Recognit.*, vol. 41, no. 8, pp. 2693–2709, 2008.

[58] H. A. Fayed and A. F. Atiya, "A novel template reduction approach for the k-nearest neighbor method," *IEEE Trans. Neural Netw.*, vol. 20, no. 5, pp. 890–896, May 2009.

[59] J. Derrac, S. García, and F. Herrera, "IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule," *Pattern Recognit.*, vol. 43, no. 6, pp. 2082–2105, 2010.

[60] P. Domingos, "Unifying instance-based and rule-based induction," *Mach. Learning*, vol. 24, no. 2, pp. 141–168, 1996.

[61] O. Luaces and A. Bahamonde, "Inflating examples to obtain rules," *Int. J. Intell. Syst.*, vol. 18, pp. 1113–1143, 2003.

[62] H. A. Fayed, S. R. Hashem, and A. F. Atiya, "Hyperspherical prototypes for pattern classification," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 23, no. 8, pp. 1549–1575, 2009.

[63] D. Wettschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," *Artif. Intell. Rev.*, vol. 11, no. 1–5, pp. 273–314, 1997.

[64] R. Paredes and E. Vidal, "Learning weighted metrics to minimize nearest-neighbor classification error," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 7, pp. 1100–1110, Jul. 2006.

[65] M. Z. Jahromi, E. Parvinnia, and R. John, "A method of learning weighted similarity function to improve the performance of nearest neighbor," *Inf. Sci.*, vol. 179, no. 17, pp. 2964–2973, 2009.

[66] C. Vallejo, J. Troyano, and F. Ortega, "InstanceRank: Bringing order to datasets," *Pattern Recognit. Lett.*, vol. 31, no. 2, pp. 133–142, 2010.

[67] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning," *Artif. Intell. Rev.*, vol. 11, pp. 11–73, 1997.

[68] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *J. Artif. Intell. Res.*, vol. 6, no. 1, pp. 1–34, 1997.

[69] R. D. Short and K. Fukunaga, "Optimal distance measure for nearest neighbor classification," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 5, pp. 622–627, Sep. 1981.

[70] C. Gagné and M. Parizeau, "Coevolution of nearest neighbor classifiers," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 21, no. 5, pp. 921–946, 2007.

[71] J. Wang, P. Neskovic, and L. Cooper, "Improving nearest neighbor rule with a simple adaptive distance measure," *Pattern Recognit. Lett.*, vol. 28, no. 2, pp. 207–213, 2007.

[72] Y. Sun, A. K. C. Wong, and M. S. Kamel, "Classification of imbalanced data: A review.," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 23, no. 4, pp. 687–719, 2009.

[73] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[74] S. García and F. Herrera, "Evolutionary under-sampling for classification with imbalanced data sets: Proposals and taxonomy," *Evol. Comput.*, vol. 17, no. 3, pp. 275–306, 2009.

[75] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.
[76] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," *Data Mining Knowl. Discov.*, vol. 17, no. 2, pp. 225–252, 2008.
[77] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2005.
[78] A. Ben-David, "A lot of randomness is hiding in accuracy," *Eng. Appl. Artif. Intell.*, vol. 20, pp. 875–885, 2007.
[79] A. Asuncion and D. Newman. (2007). *UCI machine learning repository*. [Online]. Available: http://www.ics.uci.edu/mlearn/MLRepository.html.
[80] S. García, A. Fernández, J. Luengo, and F. Herrera, "A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability," *Soft Comput.*, vol. 13, no. 10, pp. 959–977, 2009.
[81] D. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 2nd ed. London, U.K.: Chapman & Hall, 2006.
[82] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learning Res.*, vol. 7, pp. 1–30, 2006.
[83] S. García and F. Herrera, "An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons," *J. Mach. Learning Res.*, vol. 9, pp. 2677–2694, 2008.
[84] S. García, A. Fernández, J. Luengo, and 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.*, vol. 180, pp. 2044–2064, 2010.

Isaac Triguero received the M.Sc. degree in computer science from the University of Granada, Granada, Spain, in 2009, where he is currently working toward the Ph.D. degree with the Department of Computer Science and Artificial Intelligence.

His current research interests include data mining, data reduction, and evolutionary algorithms.

Joaquín Derrac received the M.Sc. degree in computer science from the University of Granada, Granada, Spain, in 2008, where he is currently working toward the Ph.D. degree with the Department of Computer Science and Artificial Intelligence.

His current research interests include data mining, data reduction, lazy learning, and evolutionary algorithms.

Salvador García 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 with the Department of Computer Science, University of Jaén, Jaén, Spain. His research interests include data mining, data reduction, data complexity, imbalanced learning, statistical inference, and evolutionary algorithms.

Francisco Herrera received the M.Sc. and Ph.D. degrees in mathematics from the University of Granada, Granada, Spain, in 1988 and 1991, respectively.

He is currently a Professor with the Department of Computer Science and Artificial Intelligence, University of Granada. He has authored or coauthored more than 150 papers in international journals. He is a coauthor of the book *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases* (Hackensack, NJ: World Scientific, 2001). He has coedited five international books and 20 special issues in international journals on different soft computing topics. He is as Associate Editor of the journals IEEE TRANSACTIONS ON FUZZY SYSTEMS, *Information Sciences*, *Mathware and Soft Computing*, *Advances in Fuzzy Systems*, *Advances in Computational Sciences and Technology*, and the *International Journal of Applied Metaheuristics Computing*. He is also an Area Editor of the journal *Soft Computing* (in the area of genetic algorithms and genetic fuzzy systems). He is also a member of several journal editorial boards, such as *Fuzzy Sets and Systems*, *Applied Intelligence*, *Knowledge and Information Systems*, *Information Fusion*, *Evolutionary Intelligence*, the *International Journal of Hybrid Intelligent Systems*, and *Memetic Computation*. His 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.