# Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study

Salvador García, Joaquín Derrac, José Ramón Cano, and Francisco Herrera

**Abstract**—The nearest neighbor classifier is one of the most used and well known techniques for performing recognition tasks. It has also demonstrated itself to be one of the most useful algorithms in data mining in spite of its simplicity. However, the nearest neighbor classifier suffers from several drawbacks such as high storage requirements, low efficiency in classification response and low noise tolerance. These weaknesses have been the subject of study for many researchers and many solutions have been proposed. Among them, one of the most promising solutions consists of reducing the data used for establishing a classification rule (training data), by means of selecting relevant prototypes. Many prototype selection methods exist in the literature and the research in this area is still advancing. Different properties could be observed in the definition of them but no formal categorization has been established yet. This paper provides a survey of the prototype selection methods proposed in the literature from a theoretical and empirical point of view. Considering a theoretical point of view, we propose a taxonomy based on the main characteristics presented in prototype selection and we analyze their advantages and drawbacks. Empirically, we conduct an experimental study involving different sizes of data sets for measuring their performance in terms of accuracy, reduction capabilities and run-time. The results obtained by all the methods studied have been verified by nonparametric statistical tests. Several remarks, guidelines and recommendations are made for the use of prototype selection for nearest neighbor classification.

**Index Terms**—Prototype selection, nearest neighbor, taxonomy, condensation, edition, classification.

## 1 Introduction

The $k$-Nearest Neighbors rule (kNN) [1] is one of the most well known and used nonparametric classifiers in Machine Learning and Data Mining (DM) tasks [2]. In spite of its simplicity, it has also demonstrated itself to be one of the most useful and effective algorithms in DM [3] and Pattern Recognition [4] and it has been considered one of the top ten methods in DM [5]. kNN is simple to implement yet powerful, owing to its theoretical properties which guarantee that for all distributions, its probability of error is bounded above by twice the Bayes probability of error. The naive implementation of this rule has no learning phase, in that it uses all the training set objects in order to classify new incoming data. Hence, it belongs to the family of lazy learners [6], [7] in opposition to the eager learners which build a parameterized compact model of the target variable [8].

Classification typically involves partitioning samples into training and testing partitions, obtaining the training set $TR$ with $N$ samples and the test set $TS$ with $M$ samples. Each sample is represented by an attribute vector, which contains a number $d$ of attributes that are quantitative or qualitative data that describe the sample. Let $\mathbf{x}_i = \{x_{i1}, x_{i2}, \ldots, x_{id}\}$ be a training sample from $TR$, $1 \leq i \leq N$ and $\mathbf{x}_j = \{x_{j1}, x_{j2}, \ldots, x_{jd}\}$ be a test sample from $TS$, $1 \leq j \leq M$, and let $\omega$ be the true class of a training sample $\mathbf{x}_i$ and $\hat{\omega}$ be the predicted class for a test sample $\mathbf{x}_j$ ($\omega, \hat{\omega} \in 1, 2, \ldots, \Omega$). Here, $\Omega$ is the total number of classes. During the training process, we use only the true class $\omega$ of each training sample to train the classifier, while during testing we predict the class $\hat{\omega}$ of each test sample. With the kNN rule, the predicted class of the test sample $\mathbf{x}_j$ is set as equal to the true class $\omega$ of the majority of the set of samples $TK$, formed by the samples $\mathbf{x}_l$ of $TR$, $1 \leq l \leq k$, when we rearrange the $TR$ set in ascending order according to the defined distance metric (in the space of samples) to $\mathbf{x}_j$. In the case of a tie, the true class is given by the closest $\mathbf{x}_l$ sample from $TK$ to $\mathbf{x}_j$ that belongs to a conflicting class.

It is well known that kNN suffers from several drawbacks [2]. Mainly, three weaknesses cause a great impact on the successful application of the algorithm. The first one is the necessity of high storage requirements in order to retain the set of examples which defines the training set and allows it to perform the decision rule. The second one is the low efficiency obtained during the computation of the decision rule, caused by multiple computations of similarities between the test and training samples. Finally, kNN (especially 1NN) presents low tolerance to noise, due to the fact that it uses all data as relevant even when the training set contains incorrect data.

Several approaches have been suggested and studied in order to tackle the drawbacks mentioned above. Increasing the kNN performance and noise tolerance is obtained by the estimation of the optimal $k$ parameter [9]

- *S. García and J.R. Cano are with the Department of Computer Science, University of Jaén, 23071, Jaén, Spain.*
  *E-mails: sglopez@ujaen.es, jrcano@ujaen.es*
- *J. Derrac and F. Herrera are with the Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain.*
  *E-mails: jderrac@decsai.ugr.es, herrera@decsai.ugr.es*



Digital Object Indentifier 10.1109/TPAMI.2011.142 0162-8828/11/$26.00 © 2011 IEEE

or making the kNN algorithm adaptive to data [10], [11] by means of determining local decision boundaries in which the shape of the neighborhoods can be modified to be more elongate if needed. The research on similarity metrics to improve the effectiveness of kNN (and other related techniques based on similarities) is very extensive in the literature [12], [13], [14], [15] together with distance functions suitable to being used under high dimensionality conditions [16]. Other techniques related to reducing computational costs involve partitioning the feature space [17], computing distances within specific nearby volumes [18], [19] or using advanced storage structures, such as k-d trees or R-trees [20]. When the samples are preprocessed into a data structure, the nearest examples can be reported efficiently. Approximate Nearest Neighbors (ANN) techniques assume that distances are measured using any class of approximation error bound, enabling to achieve significantly faster running times. They have demonstrated excellent performance in large dimension domains [21], [22].

Nevertheless, a successful technique which simultaneously faces the computational complexity, storage requirements and noise tolerance of kNN is based on data reduction. 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, they are known as reduction techniques [23], Instance Selection [24] or Prototype Selection (PS) methods [25]. A formal specification of the PS problem is the following: Let $S \subseteq TR$ be the subset of selected samples resulting from the execution of a PS algorithm, then we classify a new pattern $\mathbf{x}_j$ from $TS$ by the kNN rule acting over $S$ instead of $TR$.

PS methods select a subset of examples from the original training data. Depending on the strategy followed by the methods, they can remove noisy, redundant and both kinds of examples. The main advantage indicated in PS methods is the capacity to choose relevant examples without generating new artificial data. Many applications manage real data and the generation of new data does not make sense. The PS problem is frequently confused with other similar problems known as Prototype Generation (PG) or abstraction methods [26]. Some researchers include PG into PS, but PG methods generate and replace the original data with new artificial data [27] allowing it to fill regions in the domain of the problem which have no representative examples in original data.

A widely used categorization of PS methods consists of three types of techniques: edition methods, condensation methods and hybrid methods [24], [28]. The goal of edition methods is to remove noisy instances in order to increase classifier accuracy. Condensation methods aim to compute a training-set-consistent subset, removing superfluous instances that will not affect the classification accuracy of the training set. Finally, hybrid methods search for a small subset of the training set that simultaneously achieves the elimination of both noisy and superfluous instances.

Some reviews of PS methods can be found in the literature [23], [24], [29], [30]. However, the characteristics of the methods are not studied completely and they do not present a taxonomy which could classify all methods according to their similarities. For example, in [23], the main properties of the PS methods are analyzed but no categorization is set out; or in [29], [30], PS methods are not differentiated from PG methods.

Apart from the absence of a complete taxonomy of PS methods in the literature, we have observed that the algorithms proposed are usually compared with a subset of the complete family of PS methods and, in most of the studies, no rigorous analysis has been carried out. Furthermore, many new methods have been proposed in recent years and they are going unnoticed in respect to the PS method reviewed in well-known surveys [23], [29]. Figure 1 illustrates a comparison network where each node corresponds to a PS algorithm and a directed vertex between two nodes indicates that the algorithm of the start node has been compared with the algorithm of the end node. The size of the node is correlated to the number of input and output vertices. We can see that most of the PS algorithms are represented by small nodes and that the graph is far from being complete, which has prompted the present paper.

Dealing with large data sets is also possible with kNN when PS is applied to them. In [31], a process called stratification was proposed for PS in order to cope with large data sets, offering excellent results. In this paper, we also address the improvement achieved by the combination of stratification and PS in comparison to other alternatives based on ANN.

The mentioned reasons motivate the global purpose of this paper, which can be divided into four objectives:

- To propose a complete taxonomy based on the main properties observed using the PS methods. The taxonomy will allow us to know the advantages and drawbacks from a theoretical point of view.
- To make an empirical study for analyzing the methods in terms of accuracy, reduction capabilities and time complexity. Our goal is to identify the best methods in each family and to stress the relevant properties of each one.
- To compare the PS methods with other related techniques that speed up the kNN computation when tackling very large data sets. ANN techniques will be compared with stratified PS.
- To illustrate through graphical representations of selected data the effect of the main PS methods. Graphical representations help us to understand the results obtained in the experimental study.

The experimental study will include a statistical analysis based on nonparametric tests and we will conduct experiments involving a total of 42 PS methods and 58 small and medium size data sets. The comparison with ANN methods involve 7 large data sets more. The graphical representations of selected data will be done

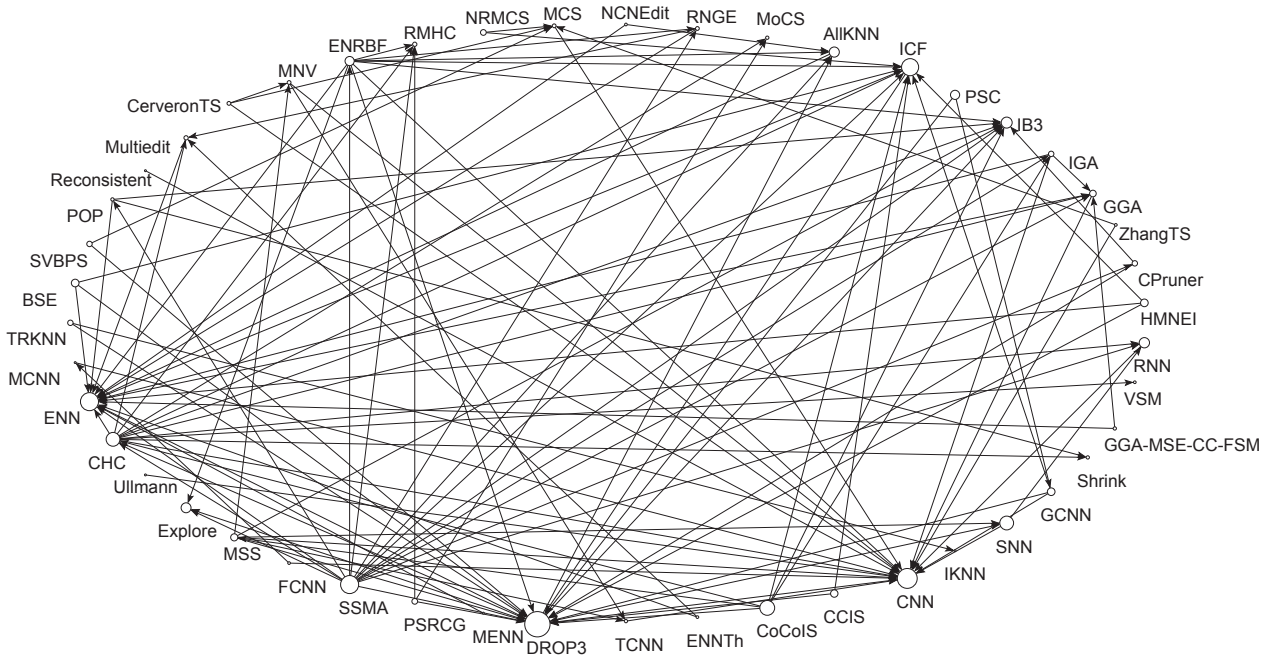



Fig. 1: Comparison Network of PS methods. Later, the methods will be defined in Table 1.

by using a 2-dimensional data set called *banana* with moderate complexity features.

This paper is organized as follows. The related and advanced work on PS is given in Section 2. Section 3 presents the PS methods reviewed, their properties and the taxonomy proposed. Section 4 describes the experimental framework for small and medium data sets, examines the results obtained in the empirical study and presents a discussion of them. The study of large data sets as well as the comparison with ANN is conducted in Section 5. Graphical representations of selected data by PS methods are illustrated in Section 6. Section 7 concludes the paper. Finally, we must point out that the paper is associated with the web page http://sci2s.ugr.es/pstax which collects extra data regarding algorithms descriptions and implementations and detailed experimental results.

## 2 Related and Advanced Work

Research in improving kNN through data preprocessing is common and in high demand nowadays. PS could represent a feasible and promising technique to obtain expected results, which justifies its relationship to other methods and problems. This section provides a wide review of other topics closely related to PS and describes other works and future trends which have been studied in the last few years.

- *Prototype Generation/Abstraction*: These methods are not limited only to select examples from the training set. They could also modify the values of the samples, changing their position in the $d$-dimensional space considered. Most of them use merging or divide and conquer strategies to set new artificial samples [32], or are based on clustering approaches [29], Learning Vector Quantization (LVQ) [33] hybrids, advanced proposals [26], [27] and evolutionary algorithms based schemes [34], [35], [36].
- *Instance and rule learning hybridizations*: This 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 data base 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 [37], [38], [39].
- *Weighting, Boosting*: This area refers to the combination of PS methods with other well-known schemes used for improving accuracy in classification problems. For example, the weighting scheme combines the PS with the Feature Selection [40], [41] or Feature Weighting [42], [43], [44], where a vector of weights associated with each attribute determines and influences the distance computations. In boosting, a PS method is run several times and a classification decision is made according to the majority class obtained over several subsets and the kNN rule [45], [46].
- *Distance Functions*: Several distance metrics have been used with kNN and PS, especially when work-

ing with categorical attributes [47]. There are some PS approaches which learn not only the subset of the selected prototype, but also the distance metric employed [48], [49]. Also, PS is suitable for use on other types of dissimilarity based classifiers [25], [50].

- *Scaling up*: One of the disadvantages of the PS methods is that most of them report a prohibitive run time or even cannot be applied over large size data sets. Recent improvements in this field cover the stratification of data [31], [51], [52] and the development of distributed approaches for PS [53].
- *Training Set Selection*: The literature includes some attempts at using instance selection to obtain subsets of examples suitable for use as an input to other DM and machine learning algorithms, such as decision trees [54] and neural networks [55]. Different problems to classification have also been dealt with using instance selection, such as subgroup discovery [56], [57] and multiple instance learning [58], [59].
- *Imbalanced learning*: One of the most promising techniques in imbalanced learning is based on data preprocessing, such as re-sampling of data mainly focused on less important concepts with respect to the minority classes [60]. It is noticeable that most of the under-sampling approaches are modifications of classic PS methods to increase the data balance [61], [62].
- *Data Complexity*: This area studies the effect on the complexity of data when PS methods are applied previous to the classification [63] or how to make a useful diagnosis of the benefits of applying PS methods taking into account the complexity of the data [64], [65].

Works and proposals enumerated in this subsection are out of the scope of this paper. We have to point out that the main objective of this paper is to give a wide overview of the PS methods proposed in the literature and to establish a comparison of them without considering external and classifier dependant factors, such as distance functions and weighting; advanced improvements for specific goals, such as improving efficiency or application to more complex domains; and advanced representations, such as rule hybridizations and prototype abstractions.

## 3 Prototype Selection Taxonomy

This section presents the taxonomy of PS methods and the criteria used for building it. First, in Subsection 3.1, the main characteristics which will define the categories of the taxonomy will be outlined. In Subsection 3.2, we briefly enumerate all the PS methods proposed in the literature. The complete and abbreviated name will be given together with the reference. Finally, Subsection 3.3, presents the taxonomy.

### 3.1 Common Properties in Prototype Selection Methods

This section provides a framework for the discussion of the PS methods presented in the next subsection. The issues discussed include order of the search, type of selection and evaluation of the search. These mentioned issues are involved in the definition of the taxonomy, since they are exclusive to the operation of the PS algorithms. Other classifier-dependent issues such as distance functions or exemplar representation will be presented. Finally, some criteria will also be pointed out in order to compare PS methods.

#### 3.1.1 Direction of Search

When searching for a subset $S$ of prototypes to keep from training set $TR$, there are a variety of directions in which the search can proceed:

- **Incremental:** An incremental search begins with an empty subset $S$, and adds each instance in $TR$ to $S$ if it fulfills some criteria. In this case, the algorithm depends on the order of presentation and this factor could be very important. Under such a scheme, the order of presentation of instances in $TR$ should be random because by definition, an incremental algorithm should be able to handle new instances as they become available without all of them being present at the beginning. Nevertheless, some recent incremental approaches are order-independent because they add instances to $S$ in a somewhat incremental fashion, but they examine all available instances to help select which instance to add next. This makes the algorithm not truly incremental as we have defined above, although we will also consider them as incremental approaches.
  One advantage of an incremental scheme is that if instances are made available later, after training is complete, they can continue to be added to $S$ according to the same criteria. This capability could be very helpful when dealing with data streams or online learning. Another advantage is that they can be faster and use less storage during the learning phase than non-incremental algorithms. The main disadvantage is that incremental algorithms must make decisions based on little information and are therefore prone to errors until more information is available.
- **Decremental:** The decremental search begins with $S = TR$, and then searches for instances to remove from $S$. Again, the order of presentation is important, but unlike the incremental process, all of the training examples are available for examination at any time.
  One disadvantage with the decremental rule is that it presents a higher computational cost than incremental algorithms. Furthermore, the learning stage must be done in an off-line fashion because decremental approaches need all possible data. However,

if the application of a decremental algorithm can result in greater storage reduction, then the extra computation during learning (which is done just once) can be well worth the computational savings during execution thereafter.

- **Batch:** Another way to apply a PS process is in batch mode. This involves deciding if each instance meets the removal criteria before removing any of them. Then all those that do meet the criteria are removed at once. As with decremental algorithms, batch processing suffers from increased time complexity over incremental algorithms.
- **Mixed:** A mixed search begins with a pre-selected subset $S$ (randomly or selected by an incremental or decremental process) and iteratively can add or remove any instance which meets the specific criterion. This type of search allows rectifications to already done operations and its main advantage is to make easy to obtain good accuracy-suited subsets of instances. It usually suffers from the same drawbacks reported in decremental algorithms, but this fact depends to a great extent on the specific proposal. Note that these kinds of algorithms are closely related to the order-independent incremental approaches but, in this case, instance removal from $S$ is allowed.
- **Fixed:** A fixed search is a subfamily of mixed search in which the number of additions and removals remains the same. Thus, the number of final prototypes is determined at the beginning of the learning phase and is never changed. This strategy of search is not very common in PS, although it is typical in PG methods, such as LVQ.

#### 3.1.2 Type of Selection

This factor is mainly conditioned by the type of search carried out by the PS algorithms, whether they seek to retain border points, central points or some other set of points.

- **Condensation:** This set includes the techniques which aim to retain the points which are closer to the decision boundaries, also called border points. The intuition 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 due to the fact that there are fewer border points than internal points in most of the data.
- **Edition:** These kinds of algorithms instead seek to remove border points. They remove points that are noisy or do not agree with their neighbors. This removes close border points, 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.
- **Hybrid:** Hybrid methods try to find the smallest subset $S$ which maintains or even increases the generalization accuracy in test data. To achieve this, it allows the removal of internal and border points based on criteria followed by the two previous strategies. The kNN classifier is highly adaptable to these methods, obtaining great improvements even with a very small subset of instances selected.

#### 3.1.3 Evaluation of Search

kNN is a simple technique and it can be used to direct the search of a PS algorithm. The objective pursued is to make a prediction on a non-definitive selection and to compare between selections. This characteristic influences the quality criterion and it can be divided into:

- **Filter:** When the kNN rule is used for partial data to determine the criteria of adding or removing and no leave-one-out validation scheme is used to obtain a good estimation of generalization accuracy. The fact of using subsets of the training data in each decision increments the efficiency of these methods, but the accuracy may not be enhanced.
- **Wrapper:** When the kNN rule is used for 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, which helps to obtain better accuracy over test data. However, each decision involves a complete computation of the kNN rule over the training set and the learning phase can be computationally expensive.

#### 3.1.4 Other properties

We can remark on other properties related to PS. They influence the operation and results which can be obtained with PS in combination with kNN. However, these properties are dependent on the type of kNN employed, or define different data reduction methods and they are not good for establishing a distinction or taxonomy among them.

- *Representation:* This issue deals with the type of examples retained in the subset $S$. In its formal definition PS methods only allow subsets of existing examples in the training set to be obtained. Other types of representation (pointed out in Section 2) could tolerate the modification of examples to represent collections of instances to form rules.
- *Distance Function:* The distance function (or similarity function) is used to decide which neighbors are closest to an input vector and can have a dramatic effect on an instance-based learning system. Two

distance functions are the most used in kNN: the Euclidean distance and the HVDM distance [47].

- *Voting:* Another decision that must be made is the choice of $k$, which is the number of neighbors used to decide the output class of an input vector. Furthermore, within the $k$ nearest neighbors of input data, ties may occur among two or more classes and a decision must also be made. In such cases, an arbitrary selection of the class or a distance-weighted choice is used.

Note that the three properties analyzed here will depend on the properties of the kNN (or instance-based learning) approach that we use. It is logical to provide the PS method with a similar distance function and voting schemes to the ones used by the subsequent kNN classifier.

#### 3.1.5 Criteria to Compare PS Methods

When comparing PS methods, there are a number of criteria that can be used to evaluate the relative strengths and weaknesses of each algorithm. These include storage reduction, noise tolerance, generalization accuracy and time requirements.

- *Storage reduction:* One of the main goals of the PS 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.
- *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. Secondly, the generalization accuracy can suffer, especially if noisy instances are retained instead of good instances.
- *Generalization accuracy:* A successful algorithm will often be able to significantly reduce the size of the training set without significantly reducing generalization accuracy.
- *Time requirements:* Usually, the learning process is done just once on a training set, so 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.

### 3.2 Prototype Selection Methods

More than 50 PS methods have been proposed in the literature. This section is devoted to enumerating and designating them according to a standard followed in this paper. For more details on their descriptions and implementations, the reader can visit the URL associated to this paper. Implementations of the algorithms in Java can be found in KEEL software [114].

Table 1 presents an enumeration of PS methods reviewed in this paper. The complete name, abbreviation

TABLE 1: PS methods reviewed

| Complete name | Abbr. name | Reference |
|---|---|---|
| Condensed Nearest Neighbor | CNN | [66] |
| Reduced Nearest Neighbor | RNN | [67] |
| Edited Nearest Neighbor | ENN | [68] |
| *No name specified* | Ullmann | [69] |
| Selective Nearest Neighbor | SNN | [70] |
| Repeated Edited Nearest Neighbor | RENN | [71] |
| **All-KNN** | **AllKNN** | |
| Tomek Condensed Nearest Neighbor | TCNN | [72] |
| Mutual Neighborhood Value | MNV | [73] |
| MultiEdit | MultiEdit | [74], [75] |
| Shrink | Shrink | [76] |
| Instance Based 2 | IB2 | [77] |
| **Instance Based 3** | **IB3** | |
| Monte Carlo 1 | MC1 | [40] |
| **Random Mutation Hill Climbing** | **RMHC** | |
| Minimal Consistent Set | MCS | [78] |
| Encoding Length Heuristic | ELH | [79] |
| Encoding Length Grow | ELGrow | |
| **Explore** | **Explore** | |
| Model Class Selection | MoCS | [80] |
| Variable Similarity Metric | VSM | [81] |
| Gabriel Graph Editing | GGE | [82] |
| **Relative Neighborhood Graph Editing** | **RNGE** | |
| Polyline Functions | PF | [83] |
| Generational Genetic Algorithm | GGA | [84], [85] |
| Modified Edited Nearest Neighbor | MENN | [86] |
| Decremental Reduction Optimization Procedure 1 | DROP1 | [23] |
| Decremental Reduction Optimization Procedure 2 | DROP2 | |
| **Decremental Reduction Optimization Procedure 3** | **DROP3** | |
| Decremental Reduction Optimization Procedure 4 | DROP4 | |
| Decremental Reduction Optimization Procedure 5 | DROP5 | |
| Decremental Encoding Length | DEL | |
| Estimation of Distribution Algorithm | EDA | [87] |
| Tabu Search | CerveronTS | [88] |
| Iterative Case Filtering | ICF | [89] |
| Modified Condensed Nearest Neighbor | MCNN | [90] |
| Intelligent Genetic Algorithm | IGA | [91] |
| Prototype Selection using Relative Certainty Gain | PSRCG | [92], [93] |
| Improved KNN | IKNN | [94] |
| Tabu Search | ZhangTS | [95] |
| Iterative Maximal Nearest Centroid Neighbor | Iterative MaxNCN | [96] |
| **Reconsistent** | **Reconsistent** | |
| C-Pruner | CPruner | [97] |
| Steady-State Genetic Algorithm | SSGA | [98] |
| Population Based Incremental Learning | PBIL | |
| **CHC Evolutionary Algorithm** | **CHC** | |
| Patterns by Ordered Projections | POP | [99] |
| Nearest Centroid Neighbor Edition | NCNEdit | [100] |
| **Edited Normalized Radial Basis Function** | **ENRBF** | [24] |
| Edited Normalized Radial Basis Function 2 | ENRBF2 | |
| Edited Nearest Neighbor Estimating Class Probabilistic | ENNProb | [101] |
| **Edited Nearest Neighbor Estimating** | **ENNTh** | [101] |
| **Class Probabilistic and Threshold** | | |
| Support Vector based Prototype Selection | SVBPS | [102] |
| Backward Sequential Edition | BSE | [103] |
| Modified Selective Subset | MSS | [104] |
| Generalized Condensed Nearest Neighbor | GCNN | [105] |
| **Fast Condensed Nearest Neighbor 1** | **FCNN** | [28] |
| Fast Condensed Nearest Neighbor 2 | FCNN2 | |
| Fast Condensed Nearest Neighbor 3 | FCNN3 | |
| Fast Condensed Nearest Neighbor 4 | FCNN4 | |
| Noise Removing based on Minimal Consistent Set | NRMCS | [106] |
| Genetic Algorithm based on Mean Square Error, | GA-MSE-CC-FSM | [107] |
| Clustered Crossover and Fast Smart Mutation | | |
| Steady-State Memetic Algorithm | SSMA | [108] |
| Hit Miss Network C | HMNC | [109] |
| Hit Miss Network Edition | HMNE | |
| **Hit Miss Network Edition Iterative** | **HMNEI** | |
| Template Reduction for KNN | TRKNN | [110] |
| Prototype Selection based on Clustering | PSC | [111] |
| Class Conditional Instance Selection | CCIS | [112] |
| Cooperative Coevolutionary Instance Selection | CoCoIS | [113] |

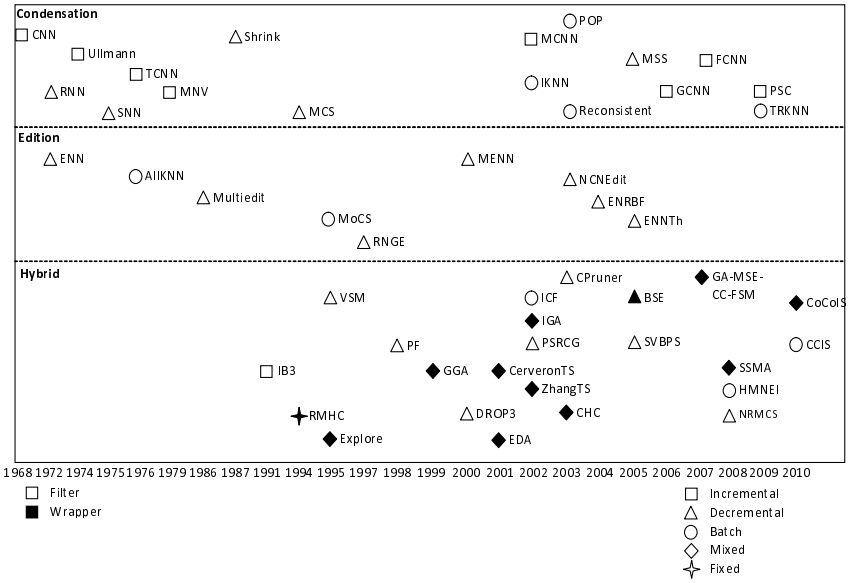



Fig. 2: Prototype Selection Map

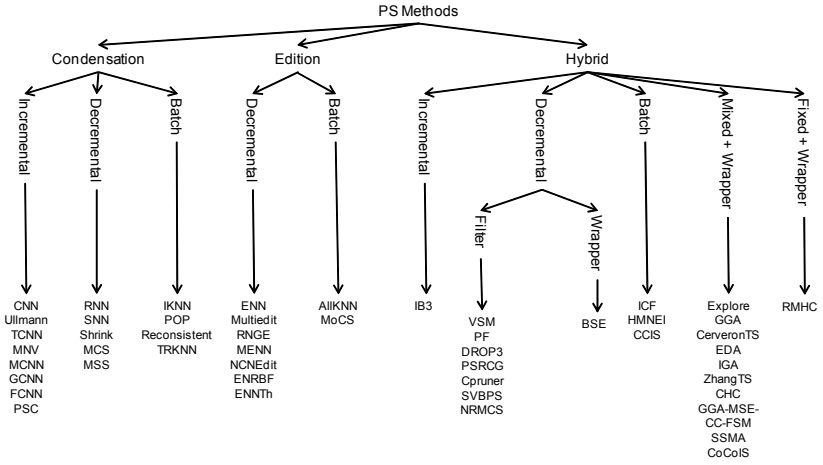



Fig. 3: Prototype Selection Taxonomy

and reference are 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.

### 3.3 Taxonomy of Prototype Selection Methods

The properties studied above can be used to categorize the PS methods proposed in the literature. The direction of the search, type of selection and evaluation of the search may differ among PS methods and constitute a set of properties which are exclusive to the way of operating of the PS methods. This section presents the taxonomy of PS methods based on these properties.

In order to situate the PS methods in time, we illustrate a map of the main methods proposed in each paper enumerated in Table 1. We refer to as main methods those which are the preferred or have reported the best results in the paper in which they were proposed (in other words, the ones in bold when more than one method is proposed in a certain paper). Figure 2 depicts the map of PS methods. The figure allows us to point out interesting facts:

- Condensation and Edition techniques display opposite behavior and they were joined when IB3 was proposed. IB3 is the first hybrid method which combines an edition stage with a condensation one. Since its proposal, there has been a significant effort in proposing new hybrid approaches, decreasing the proposals of condensation methods.
- Few edition methods have been proposed in comparison to the other two families. The main reasons are that the first edition method, ENN, obtains good results in conjunction with kNN and the edition approaches do not achieve high reduction rates, which is one of the objects of interest in PS. Incremental edition approaches have not been proposed because it is very important to know the complete set of data for identifying noisy instances.

- Recent efforts in proposing PS methods are being noted in condensation and hybrid approaches. Both of them could be made in any direction search, but the mixed direction search is typical in hybrid methods and it is not presented in condensation methods.
- Wrapper evaluation searches are only presented in hybrid approaches (usually in a mixed direction search). This evaluation search is intended to optimize a selection, without thinking of computational costs. The resulting selection depends on the whole training set whereas in edition and condensation approaches, the decision is made considering only local information.

Furthermore, Figure 3 illustrates the categorization following a hierarchy based on this order: type of selection, direction of search and evaluation of the search. It allows us to distinguish among families of methods and to estimate the size of each one.

One of the objectives of this paper is to highlight the best methods depending on their properties, taking into account that we are conscious that the properties could determine the suitability of use of a specific scheme. To do this, in Section 4, we will conclude which methods perform best for each family considering several metrics of performance.

## 4 Experimental Framework, Empirical Study and Analysis of Results: Small and Medium Data Sets

This section presents the experimental framework followed in this paper, together with the results collected and discussions on them. Subsection 4.1 will describe the complete experimental set up. Then, the study will be divided into two parts: study and analysis of the results obtained over small data sets (Subsection 4.2) and over medium data sets (Subsection 4.3). Finally, Subsection 4.4 will provide a global discussion of the results obtained.

### 4.1 Experimental Set Up

The aim of this section is to show all the factors and issues related to the experimental study. We specify the data sets, validation procedure, parameters of the algorithms, performance metrics and PS methods involved in the analysis. The statistical tests used to contrast the results are also briefly commented on the end of this section.

The performance of PS algorithms is analyzed by using 58 data sets taken from the UCI Machine Learning Database Repository [115] and KEEL data set repository [1]. Data sets are divided into two categories: small size and medium size data sets. The small size data sets have no more than 2,000 instances, whereas medium data sets have no more than 20,000 instances. Large size data sets will be considered later (in a separate study, see Section 5).

The main characteristics of these data sets are summarized in Table 2. For each data set, the name, number of examples, number of attributes (numeric and nominal) and number of classes are given.

The data sets considered are partitioned using the ten fold cross-validation (10-fcv) procedure. The parameters of the PS algorithms are those recommended by their respective authors. We assume that the choice of the values of parameters is optimally chosen by their own authors. Nevertheless, in the PS methods that require the specification of the number of neighbors as parameter, its value coincides with the $k$ value of the kNN rule afterwards. But all edition methods consider a minimum of 3 nearest neighbors to operate (as recommended in [68]), although they were applied to a 1NN classifier. The Euclidean distance is chosen as the distance metric because it is well-known and the most used for kNN. All probabilistic methods (including incremental methods which depend on the order of instance presentation) are run three times and the final results obtained correspond to the average performance values of these runs.

Two performance measures are widely used because of their simplicity and successful application when multi-class classification problems are treated. We refer to accuracy and Cohen's kappa [116] measures, which will be adopted to measure the efficacy of the PS methods in terms of classification success.

- *Accuracy*: is the number of successful hits relative to the total number of classifications. It has been by far the most commonly used metric for assessing the performance of classifiers for years [117].
- *Cohen's kappa*: is an alternative to *accuracy*, a method, known for decades, which compensates for random hits [116]. Its original purpose was to measure the degree of agreement or disagreement between two people observing the same phenomenon. Cohen's kappa can be adapted to classification tasks and its use is recommended because it takes random successes into consideration as a standard, in the same way as the AUC measure [118].
  An easy way of computing Cohen's kappa is to make use of the resulting confusion matrix in a classification task. Specifically, the Cohen's kappa measure can be obtained using the following expression:

$$kappa = \frac{N \sum_{i=1}^{\Omega} y_{ii} - \sum_{i=1}^{\Omega} y_{i.} y_{.i}}{N^2 - \sum_{i=1}^{\Omega} y_{i.} y_{.i}},$$

  where $y_{ii}$ is the cell count in the main diagonal of the resulting confusion matrix, $N$ is the number of examples, $\Omega$ is the number of class values, and $y_{.i}$,$y_{i.}$ are the columns and rows total counts of the confusion matrix, respectively. Cohen's kappa ranges from $-1$ (total disagreement) through $0$ (random classification) to $1$ (perfect agreement). Being a

1. http://www.keel.es/datasets.php

TABLE 2: Summary description for classification data sets

| Data Set | #Ex. | #Atts. | #Num. | #Nom. | #Cl. | Data Set | #Ex. | #Atts. | #Num. | #Nom. | #Cl. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| abalone | 4,174 | 8 | 7 | 1 | 28 | mammographic | 961 | 5 | 5 | 0 | 2 |
| appendicitis | 106 | 7 | 7 | 0 | 2 | marketing | 8,993 | 13 | 13 | 0 | 9 |
| australian | 690 | 14 | 8 | 6 | 2 | monk-2 | 432 | 6 | 6 | 0 | 2 |
| 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 | 303 | 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 | 34 | 34 | 0 | 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 | 6 | 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 |

scalar, it is less expressive than ROC curves when applied to binary-classification. However, for multi-class problems, kappa is a very useful, yet simple, meter for measuring the accuracy of the classifier while compensating for random successes.

The set of PS methods involved in the experimental study should be reduced for space restrictions and to avoid obtaining unnecessary results. It is determined by the following guidelines:

- One method is chosen from each proposal paper. The preferred one is that which performs best given the instructions of the corresponding authors. In the case of having two or more methods of different capabilities (i.e., efficiency vs. efficacy), we prefer the best performing in terms of efficacy. The PS methods selected, in the case of there being more than one proposal per paper, are those highlighted in bold in Table 1.
- All PS methods must have a reasonable time complexity over small data sets. Many of the proposals are unable to be run over a data set with more than 500 instances. It is the case in the following algorithms: CerveronTS, ZhangTS, BSE and GA-MSE-CC-FSM.
- Old proposals that have not had much attention in the literature do not participate in the experimental study. This is the case with: Ullmann, PF and EDA.

Thus, the empirical study involves 42 PS methods from those listed in Table 1. We want to outline that the implementations are only based on the descriptions and specifications given by the respective authors in their papers. No advanced data structures and enhancements for improving the efficiency of PS methods have been carried out. All methods (including the slowest ones) are collected in KEEL software [114].

Statistical analysis will be carried out by means of nonparametric statistical tests. In [119], [120], [121], authors recommend a set of simple, safe and robust non-parametric tests for statistical comparisons of classifiers. The Wilcoxon test [122] will be used in order to conduct pairwise comparisons among all PS methods considered in the study. The reader can also look up the web page http://sci2s.ugr.es/sicidm for more details in nonparametric statistical analysis.

### 4.2 Analysis and Empirical Results on Small Size Data Sets

Table 3 presents the average results obtained by the PS methods over the 39 small size data sets. $Red.$ denotes reduction rate achieved, $tst\ Acc.$ and $tst\ Kap.$ denote the accuracy and kappa obtained in test data, respectively; $Acc.*Red.$ and $Kap.*Red.$ correspond to the product of accuracy/kappa and reduction rate, which is an estimator of how good a PS method is considering a tradeoff of reduction and success rate of classification. Finally, $Time$ denotes the average time elapsed in seconds to

TABLE 3: Average results obtained by the PS methods over small data sets

| *Red.* | | *tst Acc.* | | *tst Kap.* | | *Acc. ∗ Red.* | | *Kap. ∗ Red.* | | *Time* | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Explore | 0.9789 | **CHC** | 0.7609 | **SSMA** | 0.5420 | **CHC** | 0.7399 | **CHC** | 0.5255 | 1NN | – |
| CHC | 0.9725 | SSMA | 0.7605 | CHC | 0.5404 | SSMA | 0.7283 | SSMA | 0.5190 | CNN | 0.0027 |
| NRMCS | 0.9683 | GGA | 0.7566 | GGA | 0.5328 | Explore | 0.7267 | GGA | 0.5014 | POP | 0.0091 |
| SSMA | 0.9576 | **RNG** | 0.7552 | **RMHC** | 0.5293 | GGA | 0.7120 | **RMHC** | 0.4772 | PSC | 0.0232 |
| GGA | 0.9411 | **RMHC** | 0.7519 | **HMNEI** | 0.5277 | **RMHC** | 0.6779 | Explore | 0.4707 | ENN | 0.0344 |
| RNN | 0.9187 | **MoCS** | 0.7489 | **RNG** | 0.5268 | **RNN** | 0.6684 | **RNN** | 0.4309 | IB3 | 0.0365 |
| CCIS | 0.9169 | ENN | 0.7488 | **MoCS** | 0.5204 | NRMCS | 0.6639 | CoCoIS | 0.4294 | MSS | 0.0449 |
| IGA | 0.9160 | NCNEdit | 0.7482 | NCNEdit | 0.5122 | IGA | 0.6434 | IGA | 0.4080 | Multiedit | 0.0469 |
| CPruner | 0.9129 | AllKNN | 0.7472 | ENN | 0.5121 | CoCoIS | 0.6281 | **MCNN** | 0.4051 | ENNTh | 0.0481 |
| MCNN | 0.9118 | **HMNEI** | 0.7436 | AllKNN | 0.5094 | **MCNN** | 0.6224 | NRMCS | 0.3836 | FCNN | 0.0497 |
| RMHC | 0.9015 | ENNTh | 0.7428 | CoCoIS | 0.4997 | **CCIS** | 0.6115 | **DROP3** | 0.3681 | MoCS | 0.0500 |
| CoCoIS | 0.8594 | Explore | 0.7424 | ENNTh | 0.4955 | **CPruner** | 0.6084 | **CCIS** | 0.3371 | MCNN | 0.0684 |
| DROP3 | 0.8235 | MENN | 0.7364 | 1NN | 0.4918 | DROP3 | 0.5761 | **IB3** | 0.3248 | MENN | 0.0685 |
| SNN | 0.7519 | 1NN | 0.7326 | **POP** | 0.4886 | **IB3** | 0.4997 | CPruner | 0.3008 | AllKNN | 0.0905 |
| ICF | 0.7160 | CoCoIS | 0.7309 | MENN | 0.4886 | **ICF** | 0.4848 | **ICF** | 0.2936 | TRKNN | 0.1040 |
| IB3 | 0.7114 | **POP** | 0.7300 | Explore | 0.4809 | PSC | 0.4569 | HMNEI | 0.2929 | CCIS | 0.1090 |
| PSC | 0.7035 | **RNN** | 0.7276 | Multiedit | 0.4758 | TCNN | 0.4521 | TCNN | 0.2920 | HMNEI | 0.1234 |
| SVBPS | 0.6749 | Multiedit | 0.7270 | **MSS** | 0.4708 | FCNN | 0.4477 | FCNN | 0.2917 | ENRBF | 0.1438 |
| Shrink | 0.6675 | MSS | 0.7194 | RNN | 0.4691 | SVBPS | 0.4448 | MNV | 0.2746 | PSRCG | 0.1466 |
| TCNN | 0.6411 | **FCNN** | 0.7069 | **FCNN** | 0.4605 | SNN | 0.4324 | CNN | 0.2631 | CPruner | 0.1639 |
| FCNN | 0.6333 | MCS | 0.7060 | **IB3** | 0.4566 | MNV | 0.4266 | SVBPS | 0.2615 | ICF | 0.1708 |
| MNV | 0.6071 | CNN | 0.7057 | CNN | 0.4560 | HMNEI | 0.4128 | PSC | 0.2594 | Shrink | 0.1811 |
| CNN | 0.5771 | TCNN | 0.7052 | MCS | 0.4559 | CNN | 0.4072 | **MENN** | 0.2443 | VSM | 0.1854 |
| VSM | 0.5669 | IKNN | 0.7027 | TCNN | 0.4555 | **Reconsistent** | 0.3840 | **Reconsistent** | 0.2406 | IKNN | 0.1920 |
| Reconsistent | 0.5581 | MNV | 0.7026 | MNV | 0.4523 | **MENN** | 0.3682 | MCS | 0.2348 | NRMCS | 0.2768 |
| HMNEI | 0.5551 | **IB3** | 0.7024 | IKNN | 0.4494 | MCS | 0.3637 | ENNTh | 0.2294 | NCNEdit | 0.3674 |
| TRKNN | 0.5195 | IGA | 0.7024 | **DROP3** | 0.4470 | VSM | 0.3600 | TRKNN | 0.2077 | MCS | 0.4126 |
| MCS | 0.5151 | **DROP3** | 0.6997 | IGA | 0.4455 | TRKNN | 0.3496 | MSS | 0.2073 | DROP3 | 0.5601 |
| PSRCG | 0.5065 | Reconsistent | 0.6880 | MCNN | 0.4443 | ENNTh | 0.3439 | PSRCG | 0.2072 | SNN | 0.7535 |
| MENN | 0.5000 | NRMCS | 0.6856 | Reconsistent | 0.4310 | PSRCG | 0.3433 | SNN | 0.1983 | SVBPS | 1.0064 |
| ENNTh | 0.4629 | ENRBF | 0.6837 | ICF | 0.4101 | Shrink | 0.3411 | VSM | 0.1964 | TCNN | 1.9487 |
| GCNN | 0.4542 | MCNN | 0.6826 | PSRCG | 0.4092 | MSS | 0.3168 | **AllKNN** | 0.1799 | Explore | 2.1719 |
| MSS | 0.4404 | PSRCG | 0.6779 | TRKNN | 0.3999 | GCNN | 0.3022 | GCNN | 0.1774 | MNV | 2.5741 |
| AllKNN | 0.3532 | ICF | 0.6772 | NRMCS | 0.3962 | **AllKNN** | 0.2639 | Multiedit | 0.1657 | Reconsistent | 4.5228 |
| Multiedit | 0.3483 | TRKNN | 0.6729 | GCNN | 0.3905 | Multiedit | 0.2532 | IKNN | 0.1444 | RNG | 7.1695 |
| IKNN | 0.3214 | CCIS | 0.6669 | SVBPS | 0.3875 | IKNN | 0.2258 | ENN | 0.1293 | RNN | 16.1739 |
| ENRBF | 0.3042 | CPruner | 0.6664 | PSC | 0.3687 | ENRBF | 0.2080 | RNG | 0.1243 | CHC | 23.7252 |
| ENN | 0.2525 | GCNN | 0.6654 | CCIS | 0.3676 | ENN | 0.1891 | Shrink | 0.1152 | SSMA | 27.4869 |
| RNG | 0.2360 | SVBPS | 0.6591 | VSM | 0.3465 | RNG | 0.1782 | NCNEdit | 0.1146 | RMHC | 32.2845 |
| NCNEdit | 0.2237 | PSC | 0.6495 | ENRBF | 0.3309 | NCNEdit | 0.1674 | ENRBF | 0.1007 | GCNN | 61.4989 |
| MoCS | 0.1232 | VSM | 0.6350 | CPruner | 0.3295 | MoCS | 0.0923 | MoCS | 0.0641 | GGA | 84.9042 |
| POP | 0.0762 | SNN | 0.5751 | SNN | 0.2638 | POP | 0.0556 | POP | 0.0372 | IGA | 122.1011 |
| 1NN | – | Shrink | 0.5110 | Shrink | 0.1726 | 1NN | – | 1NN | – | CoCoIS | 267.3500 |

**It cannot be run over medium data sets for efficiency reasons*

complete a run of a PS method [2]. In the case of 1NN, the time required is not displayed due to the fact that no PS stage is run before. For each type of result, the algorithms are ordered from the best to the worst. Algorithms highlighted in bold are those which obtain the best result in their corresponding family, according to the taxonomy illustrated in Figure 3. They will make up the experimental study of medium size data sets, showed in the next subsection.

All detailed results for each data set and PS method (including averages and standard deviations), together with the study of the 3NN classifier can be seen at URL http://sci2s.ugr.es/pstax. In the interest of compactness, the study corresponding to 3NN has been not included in the paper mainly due to the fact that the results obtained are very similar to 1NN. They can be found at the URL given above.

The Wilcoxon test [122], [119], [120] is adopted considering a level of significance of $\alpha = 0.1$. Table 4 shows a summary of all the possible comparisons employing the Wilcoxon test among all PS methods over small data sets. This table collects the statistical comparisons of the four main performance measures used in this paper: $tst\ Acc.$, $tst\ Kap.$, $Acc. * Red.$ and $Kap. * Red.$. The individual comparisons between all possible PS methods are exhibited in the URL associated with this paper (http://sci2s.ugr.es/pstax). Table 4 shows, for each method in the row, the number of PS methods outperformed by using the Wilcoxon test under the column represented by

2. The machine used was an Intel Core i7 CPU 920 at 2.67GHz with 4GB of RAM

TABLE 4: Wilcoxon test results over small data sets

| | tst Acc. | | tst Kap. | | Acc. * Red. | | Kappa. * Red. | |
|---|---|---|---|---|---|---|---|---|
| | + | ± | + | ± | + | ± | + | ± |
| AllKNN | 25 | 40 | 22 | 40 | 7 | 16 | 10 | 25 |
| CCIS | 4 | 21 | 2 | 25 | 30 | 36 | 26 | 35 |
| **CHC** | 31 | **41** | 29 | **41** | **41** | **41** | 38 | **41** |
| CNN | 8 | 19 | 7 | 27 | 12 | 20 | 15 | 27 |
| CoCoIS | 22 | 37 | 20 | 38 | 29 | 36 | 27 | 38 |
| CPruner | 8 | 25 | 0 | 21 | 28 | 34 | 1 | 26 |
| DROP3 | 7 | 28 | 5 | 31 | 29 | 32 | 29 | 35 |
| ENN | 24 | 39 | 22 | 38 | 3 | 9 | 6 | 20 |
| **ENNTh** | 22 | **41** | 19 | **41** | 10 | 27 | 13 | 27 |
| ENRBF | 20 | 37 | 0 | 29 | 3 | 13 | 0 | 7 |
| Explore | 22 | 38 | 11 | 35 | 38 | 40 | 33 | 40 |
| FCNN | 5 | 20 | 4 | 26 | 14 | 25 | 13 | 28 |
| GCNN | 4 | 27 | 5 | 31 | 2 | 14 | 1 | 14 |
| GGA | 27 | 40 | 25 | 40 | 37 | 39 | 38 | 40 |
| **HMNEI** | 25 | 39 | 22 | **41** | 9 | 27 | 15 | 30 |
| IB3 | 5 | 23 | 5 | 29 | 23 | 29 | 22 | 31 |
| ICF | 4 | 21 | 2 | 24 | 22 | 28 | 18 | 30 |
| IGA | 7 | 25 | 5 | 28 | 30 | 34 | 32 | 35 |
| IKNN | 11 | 29 | 11 | 32 | 1 | 11 | 2 | 12 |
| MCNN | 2 | 15 | 5 | 24 | 29 | 34 | 29 | 34 |
| MCS | 7 | 23 | 9 | 29 | 16 | 28 | 13 | 30 |
| **MENN** | 27 | **41** | 21 | **41** | 11 | 27 | 12 | 29 |
| MNV | 6 | 20 | 6 | 26 | 14 | 26 | 16 | 29 |
| **ModelCS** | 26 | 39 | 26 | **41** | 1 | 2 | 1 | 8 |
| MSS | 17 | 29 | 17 | 32 | 2 | 12 | 4 | 20 |
| Multiedit | 23 | 35 | 7 | 34 | 7 | 15 | 7 | 18 |
| **NCNEdit** | 26 | 40 | 27 | **41** | 2 | 9 | 3 | 18 |
| NRMCS | 6 | 28 | 2 | 28 | 36 | 38 | 25 | 38 |
| POP | 16 | 33 | 19 | 38 | 0 | 0 | 0 | 4 |
| PSC | 0 | 10 | 3 | 11 | 17 | 26 | 9 | 27 |
| PSRCG | 5 | 15 | 4 | 19 | 5 | 16 | 2 | 20 |
| Reconsistent | 4 | 16 | 4 | 22 | 12 | 18 | 8 | 24 |
| RMHC | 27 | 38 | 26 | 40 | 33 | 37 | 34 | 39 |
| **RNG** | **34** | **41** | 29 | **41** | 3 | 9 | 5 | 18 |
| RNN | 15 | 30 | 7 | 30 | 33 | 36 | 33 | 37 |
| Shrink | 0 | 5 | 0 | 2 | 7 | 22 | 0 | 13 |
| SNN | 0 | 4 | 1 | 5 | 15 | 26 | 2 | 27 |
| **SSMA** | 28 | **41** | **30** | **41** | 39 | 40 | **39** | **41** |
| SVBPS | 2 | 17 | 3 | 24 | 18 | 27 | 12 | 27 |
| TCNN | 8 | 24 | 5 | 27 | 15 | 24 | 16 | 27 |
| TRKNN | 2 | 17 | 3 | 24 | 11 | 22 | 5 | 21 |
| VSM | 1 | 8 | 2 | 13 | 6 | 17 | 2 | 16 |

'+' symbol. The column with the '±' symbol indicates the number of wins and ties obtained by the method in the row. The maximum value for each column is highlighted in bold.

Observing Tables 3 and 4, we can point out the best performing PS methods:

- In condensation incremental approaches, all methods are very similar in behavior, except PSC, which obtains the worst results. FCNN could be highlighted in accuracy/kappa performance and MCNN with respect to reduction rate with a low decrease in efficacy.
- Two methods can be stressed from the condensation decremental family: RNN and MSS. RNN obtains good reduction rates and accuracy/kappa performances, whereas MSS also offers good performance. RNN has the drawback of being quite slow.
- In general, the best condensation methods in terms of efficacy are the decremental ones, but they have as their main drawback that they require more computation time. POP and MSS methods are the best performing in terms of accuracy/kappa, although the reduction rates are low, especially that one achieved by POP. However, no condensation method is more accurate than 1NN.
- With respect to edition decremental approaches, few differences can be observed. ENN, RNGE and NCNEdit obtain the best results in accuracy/kappa and MENN and ENNTh offers a good tradeoff considering the reduction rate. Multiedit and ENRBF are not on a par with their competitors and they are below 1NN in terms of accuracy.
- AllKNN and MoCS, in edition batch approaches, achieve similar results to the methods belonging to the decremental family. AllKNN achieves better reduction rates.
- Within the hybrid decremental family, three methods deserve mention: DROP3, CPruner and NRMCS. The latter one is the best of them, but curiously, its time complexity rapidly increases in the presence of larger data sets and it cannot tackle medium size data sets. DROP3 is more accurate than CPruner, which achieves higher reduction rates.
- Considering the hybrid mixed+wrapper methods, SSMA and CHC techniques achieve the best results.
- Remarkable methods belonging to the hybrid family are DROP3, CPruner, HMNEI, CCIS, SSMA, CHC and RMHC. Wrapper based approaches are slower.
- The global best methods in terms of accuracy or kappa are MoCS, RNGE and HMNEI.
- The global best methods considering the tradeoff reduction-accuracy/kappa are RMHC, RNN, CHC, Explore and SSMA.

### 4.3 Analysis and Empirical Results on Medium Size Data Sets

This section presents the study and analysis of medium size data sets and the best PS methods per family, which are those highlighted in bold in Table 3. The goal pursued is to study the effect of scaling up the data in PS methods. Table 5 shows the average results obtained in the distinct performance measures considered (it follows the same format as Table 3) and Table 6 summarizes the Wilcoxon test results over medium data sets.

We can analyze several details from the results collected in Tables 5 and 6:

TABLE 5: Average results obtained by the best PS methods per family over medium data sets

| *Red.* | | *tst Acc.* | | *tst Kap.* | | *Acc. ∗ Red.* | | *Kap. ∗ Red.* | | *Time* | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| MCNN | 0.9914 | RMHC | 0.8306 | RMHC | 0.6493 | SSMA | 0.8141 | SSMA | 0.6328 | 1NN | – |
| CHC | 0.9914 | SSMA | 0.8292 | SSMA | 0.6446 | CHC | 0.8018 | CHC | 0.6006 | POP | 0.1706 |
| SSMA | 0.9817 | RNG | 0.8227 | HMNEI | 0.6397 | RNN | 0.7580 | RMHC | 0.5844 | CNN | 1.1014 |
| CCIS | 0.9501 | HMNEI | 0.8176 | ModelCS | 0.6336 | RMHC | 0.7476 | RNN | 0.5617 | FCNN | 3.2733 |
| RNN | 0.9454 | ModelCS | 0.8163 | RNG | 0.6283 | GGA | 0.7331 | GGA | 0.5513 | MCNN | 4.4177 |
| GGA | 0.9076 | CHC | 0.8088 | 1NN | 0.6181 | CCIS | 0.6774 | IB3 | 0.4615 | IB3 | 6.6172 |
| RMHC | 0.9001 | GGA | 0.8078 | POP | 0.6143 | CPruner | 0.6756 | FCNN | 0.4588 | MSS | 7.9165 |
| DROP3 | 0.8926 | 1NN | 0.8060 | MSS | 0.6126 | MCNN | 0.6748 | DROP3 | 0.4578 | CCIS | 12.4040 |
| CPruner | 0.8889 | AllKNN | 0.8052 | GGA | 0.6074 | DROP3 | 0.6635 | CPruner | 0.4555 | ModelCS | 15.4658 |
| ICF | 0.8037 | POP | 0.8037 | CHC | 0.6058 | IB3 | 0.6144 | CCIS | 0.5579 | AllKNN | 24.6167 |
| IB3 | 0.7670 | RNN | 0.8017 | FCNN | 0.6034 | FCNN | 0.6052 | CNN | 0.4410 | HMNEI | 28.9782 |
| FCNN | 0.7604 | IB3 | 0.8010 | IB3 | 0.6018 | CNN | 0.5830 | MCNN | 0.4295 | CPruner | 35.3761 |
| CNN | 0.7372 | MSS | 0.8008 | CNN | 0.5982 | ICF | 0.5446 | Reconsistent | 0.3654 | MENN | 37.1231 |
| Reconsistent | 0.6800 | FCNN | 0.7960 | AllKNN | 0.5951 | Reconsistent | 0.5101 | MSS | 0.3513 | ICF | 93.0212 |
| MSS | 0.5735 | CNN | 0.7909 | RNN | 0.5941 | MSS | 0.4592 | HMNEI | 0.3422 | DROP3 | 160.0486 |
| HMNEI | 0.5350 | MENN | 0.7840 | MENN | 0.5768 | HMNEI | 0.4374 | ICF | 0.3337 | Reconsistent | 1,621.7693 |
| MENN | 0.3144 | CPruner | 0.7600 | Reconsistent | 0.5373 | MENN | 0.2465 | MENN | 0.1814 | RNG | 1,866.7751 |
| AllKNN | 0.2098 | Reconsistent | 0.7501 | DROP3 | 0.5129 | AllKNN | 0.1689 | AllKNN | 0.1248 | SSMA | 6,306.6313 |
| RNG | 0.1161 | DROP3 | 0.7433 | CPruner | 0.5124 | RNG | 0.0955 | RNG | 0.0729 | CHC | 6,803.7974 |
| POP | 0.0820 | CCIS | 0.7130 | CCIS | 0.4714 | POP | 0.0659 | POP | 0.0504 | RMHC | 12,028.3811 |
| ModelCS | 0.0646 | MCNN | 0.6806 | MCNN | 0.4332 | ModelCS | 0.0527 | ModelCS | 0.0409 | GGA | 21,262.6911 |
| 1NN | – | ICF | 0.6776 | ICF | 0.4152 | 1NN | – | 1NN | – | RNN | 24,480.0439 |

TABLE 6: Wilcoxon test results over medium data sets

| | tst Acc. | | tst Kap. | | Acc. * Red. | | Kappa. * Red. | |
|---|---|---|---|---|---|---|---|---|
| | + | ± | + | ± | + | ± | + | ± |
| AllKNN | 9 | 19 | 10 | 19 | 3 | 4 | 3 | 6 |
| CCIS | 1 | 7 | 0 | 4 | 11 | 18 | 4 | 14 |
| **CHC** | 5 | **20** | 5 | 19 | **19** | **20** | 15 | **20** |
| CNN | 4 | 10 | 5 | 13 | 6 | 11 | 7 | 15 |
| CPruner | 3 | 14 | 0 | 5 | 8 | 14 | 2 | 15 |
| DROP3 | 2 | 11 | 2 | 10 | 9 | 14 | 8 | 15 |
| FCNN | 4 | 12 | 5 | 14 | 6 | 16 | 9 | 17 |
| GGA | 5 | 13 | 4 | 14 | 12 | 18 | 12 | 17 |
| **HMNEI** | 10 | 19 | 12 | **20** | 5 | 10 | 5 | 14 |
| IB3 | 2 | 11 | 4 | 9 | 9 | 17 | 9 | 16 |
| ICF | 0 | 4 | 0 | 7 | 6 | 11 | 3 | 11 |
| MCNN | 0 | 2 | 0 | 4 | 10 | 17 | 7 | 18 |
| MENN | 11 | 19 | 8 | 18 | 4 | 8 | 4 | 9 |
| **ModelCS** | 10 | **20** | 12 | **20** | 1 | 1 | 0 | 3 |
| MSS | 6 | 18 | 9 | 18 | 3 | 10 | 4 | 11 |
| **POP** | 7 | **20** | 10 | **20** | 0 | 0 | 0 | 2 |
| Reconsistent | 1 | 10 | 3 | 10 | 3 | 9 | 4 | 11 |
| RMHC | 11 | 19 | 9 | 19 | 13 | 18 | 14 | 19 |
| **RNG** | **15** | **20** | **15** | **20** | 2 | 3 | 0 | 4 |
| RNN | 4 | 14 | 4 | 12 | 14 | 18 | 15 | 19 |
| **SSMA** | 8 | **20** | 9 | 19 | **19** | **20** | **19** | **20** |

- Five techniques outperform 1NN in terms of accuracy/kappa over medium data sets: RMHC, SSMA, HMNEI, MoCS and RNGE. Two of them are edition schemes (MoCS and RNGE) and the rest are hybrid schemes. Again, no condensation method is more accurate than 1NN.
- Some methods present clear differences when dealing with larger data sets. This is the case with AllKNN, MENN and CHC. The first two, tend to try new reduction passes in the edition process, which is against the interests of accuracy and kappa, and in medium size problems this fact is more noticeable. Furthermore, CHC loses the balance between reduction and accuracy when data size increases, due to the fact that the reduction objective becomes more easy.
- There are some techniques whose run could be prohibitive when the data scales up. This is the case for RNN, RMHC, CHC and SSMA.
- The best methods in terms of accuracy or kappa are RNGE and HMNEI.
- The best methods considering the tradeoff reduction-accuracy/kappa are RMHC, RNN and SSMA.

### 4.4 Global View of the Obtained Results

Assuming the results obtained, several PS methods could be emphasized according to the accuracy/kappa obtained (RMHC, SSMA, HMNEI, RNGE), the reduction rate achieved (SSMA, RNN, CCIS) and computational cost required (POP, FCNN). However, we want to remark that the choice of a certain method depends on various factors and the results are offered here with the intention of being useful in making this decision. For example, an edition scheme will usually outperform the standard kNN classifier in the presence of noise, but few instances will be removed. This fact could determine whether the method is suitable or not to be applied over larger data sets, taking into account the expected size of the resulting subset. We have seen that the PS methods which allow high reduction rates while preserving accuracy are usually the slowest ones (hybrid mixed approaches such as SSMA) and they may require an ad-

vanced mechanism to be applied over large size data sets or they may even be useless under these circumstances. Fast methods that achieve high reduction rates are the condensation approaches, but we have seen that they are not able to improve kNN in terms of accuracy. In short, each method has advantages and disadvantages and the results offered in this section allow the making of an informed decision within each category.

In short, and focusing on the objectives usually considered in the use of PS algorithms, we can suggest the followinf, to choose the proper PS algorithm:

- For the tradeoff reduction-accuracy rate: The algorithms which obtain the best behavior are RMHC and SSMA. However, these methods achieve a significant improvement in the accuracy rate due to a high computation cost. The methods that harm the accuracy at the expense of a great reduction of time complexity are DROP3 and CCIS.
- If the interest is the accuracy rate: In this case, the best results are to be achieved with the RNGE as editor and HMNEI as hybrid method.
- When the key factor is the condensation: FCNN is the highlighted one, being one of the fastest.

## 5 Experimental Framework, Empirical Study and Analysis of Results: Large Data Sets

This second experimental study is focused on the analysis of the behavior of PS when they tackle large problems. Since the immediate application of these methods over large sets should be avoided due to their computational cost, we will introduce the use of the stratification procedure (Section 5.1) to mitigate this drawback, and thus develop a suitable approach to large problems. We compare this approach with two well-known ANN proposals (Section 5.2) through an empirical study with several large data sets (Section 5.3). The results achieved are reported and discussed in Section 5.4.

### 5.1 Stratification

The stratification strategy [31] splits the training data into disjoint strata with equal class distribution. The initial data set is divided into two sets, $TR$ and $TS$, as usual (e.g. a tenth of the data for $TS$, and the rest for $TR$ in 10-fold cross validation). Then, $TR$ is divided into $t$ disjoint sets $TD_j$ , strata of equal size, $TD_1, TD_2 \cdots TD_t$, maintaining class distribution within each subset. In this manner, the subsets $TR$ and $TS$ can be represented as follows:

$$TR = \bigcup_{j=1}^{t} TD_j, \quad TS = TD \backslash TR$$

Then, a PS method should be applied to each $TD_j$, obtaining a selected subset $TDS_j$ for each partition. The final prototype selected set is obtained joining every $TDS_j$ obtained. Finally, the kNN classifier can be applied to $TS$, using the final prototype selected set as training data.

The use of the stratification allows us to run any PS method over reduced versions of the entire training set, thus easing the problem of dealing with very large training sets by reducing the number of instances that the PS must handle simultaneously.

### 5.2 Approximated Nearest Neighbors

Two well-known ANN approaches will be used as comparisons in this study: Balanced Box Decomposition Tree (BBD-Tree) [21] and Locality Sensitive Hashing (LSH) [22]:

- **BBD-Trees** are an improved version of the well-known k-d trees [20] which consists of two types of nodes: *split nodes* and *shrink nodes*. *Split nodes* represent partitions made by using an axis-orthogonal line to split the node, whereas *shrink nodes* denote partitions made by using a box rather than a line. By alternating *split nodes* and *shrink nodes*, both the geometric size and the number of points associated with each node are greatly reduced, thus improving the efficiency of the tree regarding both storage requirements and query time.
- **LSH** is a family of methods which share the objective of hashing the instances of the training set by using several hashing functions, which ensures that the probability of collision is much higher for instances that are close to each other than for those that are apart. With the use of these hash tables, the LSH methods are able to obtain excellent query times, even in high dimensional problems. Several types of hashing functions have been defined within the LSH framework, in order to adjust the method to the distance space defined (Hamming distance, Euclidean distance, etc.).

### 5.3 Experimental Framework

The performance of PS and ANN algorithms is analyzed by using 7 large data sets taken from the UCI Machine Learning Database Repository [115] and KEEL data set repository (see Table 7). The performance measures analyzed are the same that were employed in the former study, excepting the running time, which is split into three categories:

- Model time: Time elapsed when applying the PS method over all strata, or when using the ANN method to build the necessary data structures to efficiently answer the queries (trees, hash tables, etc.).
- Training time: Time elapsed classifying the full training set.
- Training time: Time elapsed classifying the full test set.

Regarding PS methods, CCIS, DROP3, FCNN, HMNEI, RMHC, RNG and SSMA were selected since they

TABLE 8: Average results obtained

| $Red.$ | | $tst\ Acc.$ | | $tst\ Kap.$ | | $Acc. * Red.$ | | $Kap. * Red.$ | |
|---|---|---|---|---|---|---|---|---|---|
| SSMA | 0.9844 | RNG | 0.8198 | SSMA | 0.5654 | SSMA | 0.8069 | SSMA | 0.5596 |
| CCIS | 0.9210 | SSMA | 0.8173 | HMNEI | 0.5605 | CCIS | 0.7458 | CCIS | 0.4970 |
| RMHC | 0.9006 | RMHC | 0.8133 | LSH | 0.5433 | RMHC | 0.7324 | DROP3 | 0.4870 |
| DROP3 | 0.8844 | 1-NN | 0.8060 | 1-NN | 0.5426 | DROP3 | 0.7186 | RMHC | 0.4836 |
| FCNN | 0.6956 | HMNEI | 0.8009 | RMHC | 0.5370 | FCNN | 0.5842 | FCNN | 0.4208 |
| HMNEI | 0.6008 | LSH | 0.8001 | BBDTree | 0.5346 | HMNEI | 0.4928 | HMNEI | 0.3389 |
| RNG | 0.2090 | DROP3 | 0.7952 | DROP3 | 0.5244 | RNG | 0.1430 | RNG | 0.0648 |
| BBDTree | – | CCIS | 0.7951 | CCIS | 0.5233 | BBDTree | – | BBDTree | – |
| LSH | – | BBDTree | 0.7940 | RNG | 0.5223 | LSH | – | LSH | – |
| 1-NN | – | FCNN | 0.7770 | FCNN | 0.5095 | 1-NN | – | 1-NN | – |

TABLE 7: Summary description for large classification data sets

| Data Set | #Ex. | #Atts. | #Num. | #Nom. | #Cl. | #t |
|---|---|---|---|---|---|---|
| adult | 48,842 | 14 | 6 | 8 | 2 | 4 |
| census | 299,285 | 41 | 13 | 28 | 2 | 27 |
| connect-4 | 67,557 | 42 | 0 | 42 | 3 | 6 |
| fars | 100,968 | 29 | 5 | 24 | 8 | 9 |
| kddcup | 494,020 | 41 | 26 | 15 | 23 | 45 |
| poker | 1,025,010 | 10 | 10 | 0 | 10 | 92 |
| shuttle | 58,000 | 9 | 9 | 0 | 7 | 5 |

showed several interesting capabilities in the study with medium size data sets (highest accuracy, faster running times, high reduction rates, etc.). We used the same set up for them as that used in the former study, and set up the strata size as near as possible to 10,000 instances (Table 7 indicates the exact number of strata used for each data set under the column denoted by #t). BBDTrees implementation was adapted from the one offered at http://www.cs.umd.edu/~mount/ANN/ and LSH implementation was adapted from the LSH kit available at http://lshkit.sourceforge.net/. Finally, 1NN behavior has also been analyzed as a baseline method for this study. As before, further details of the concrete set up used can be seen at URL http://sci2s.ugr.es/pstax.

### 5.4 Results and analysis

Table 8 presents the average results obtained by the PS and ANN methods over the 7 large size data sets. As before, $Red.$ denotes reduction rate achieved, $tst\ Acc.$ and $tst\ Kap.$ denote the accuracy and kappa obtained in test data, respectively; $Acc. * Red.$ and $Kap. * Red.$ correspond to the product of accuracy/kappa and reduction rate, which is an estimator of how good a PS method is considering a tradeoff of reduction and success rate of classification. In addition, Table 9 shows the statistical significances for SSMA expressed by $p$-values computed by the Wilcoxon test, the methods outperformed considering $\alpha = 0.1$ are depicted in bold. In the case of ANN methods, the measures that requires the computation of reduction capabilities ($Red.$, $Acc. * Red.$ and $Kap.*Red.$) are not specified because they do not remove any instance from the data. Instead, we can compare the

TABLE 9: Wilcoxon's statistical significances reported for SSMA ($p$-values)

| SSMA | $tst\ Acc.$ | $tst\ Kap.$ | $Acc. * Red.$ | $Kap. * Red.$ |
|---|---|---|---|---|
| **vs. DROP3** | 0.031 | 0.031 | 0.016 | 0.016 |
| **vs. FCNN** | 0.078 | 0.078 | 0.016 | 0.016 |
| **vs. RHMC** | 0.016 | 0.016 | 0.016 | 0.016 |
| vs. RNG | 1.000 | 0.205 | 0.016 | 0.016 |
| vs. HMNEI | 0.047 | 0.271 | 0.016 | 0.016 |
| **vs. 1NN** | 0.078 | 0.078 | – | – |
| **vs. BBDTree** | 0.078 | 0.094 | – | – |
| **vs. LSH** | 0.078 | 0.078 | – | – |

TABLE 10: Average time results obtained (seconds)

| $modelTime$ | | $tra\ Time$ | | $tst\ Time$ | |
|---|---|---|---|---|---|
| 1NN | 0 | SSMA | 285 | SSMA | 38 |
| LSH | 4 | LSH | 1,091 | LSH | 120 |
| BBDTree | 57 | BBDTree | 1,181 | BBDTree | 130 |
| HMNEI | 80 | CCIS | 1,240 | CCIS | 133 |
| FCNN | 100 | RMHC | 1,306 | RMHC | 155 |
| CCIS | 1,349 | DROP3 | 1,471 | DROP3 | 159 |
| RNG | 14,635 | FCNN | 4,469 | FCNN | 497 |
| DROP3 | 16,899 | RNG | 7,738 | RNG | 846 |
| SSMA | 45,193 | HMNEI | 8,003 | HMNEI | 866 |
| RMHC | 77,260 | 1NN | 27,087 | 1NN | 3,088 |

time elapsed on each type of operation. Table 10 presents the average time elapsed in seconds, where $modelTime$ denotes the time spent by the method in its *building model* phase (i.e performing of PS processes over all strata for stratified PS methods, or building trees or hash tables for ANN methods), $tra\ Time$ denotes the time elapsed in the classification of the training set, and $tst\ Time$ denotes the time elapsed in the classification of the test set [3]. As before, all detailed results for each data set and PS or ANN methods (including averages, standard deviations and statistical significances with Wilcoxon's test) can be seen at URL http://sci2s.ugr.es/pstax.

Observing Tables 8, 9 and 10, we can point out the following about the performance of stratified PS methods:

- DROP3, RMHC, CCIS and SSMA shows the best reduction power of the PS methods considered. Furthemore, the use of the stratification strategy has

3. The machine used was the same as the previous study

not harmed the reduction power of PS methods, in general, which suggests an advantage in using it when aiming to obtain very reduced subsets when tackling large problems.
- RNG, SSMA and RMHC are able to outperform 1NN in accuracy. The rest of the PS methods achieve similar behavior to 1NN.
- With respect to kappa measure, SSMA and HMNEI show the best average results. Again, most of the PS methods achieve competitive results when compared with 1NN.
- Regarding composite performance measures, $Acc.*Red.$ and $Kap.*Red.$, SSMA shows the best behavior in the study. Furthermore, CCIS, DROP3 and RMHC can be highlighted as very competitive methods when analyzing both composite measures, whereas HMNEI and RNG achieve poor results, mainly due their low reduction power.
- HMNEI and FCNN shows a very good performance regarding time elapsed in the PS phase. By constrast, DROP3, RMHC and SSMA are the slowest methods in this phase, which again highlights the importance of employing the stratification procedure in order to properly apply these methods to large problems.
- With respect to the time elapsed in training and test phases, those methods with the highest reduction power (DROP3, RMHC, CCIS and, especially, SSMA) show the best results. Furthermore, all PS methods are able to improve at least three times (nearly 100 times in the case of SSMA) the time consumption of the 1NN classifier.

In general, the behavior of the PS methods when combined with the stratification procedure has been shown to be satisfactory. When facing a given large problem, a practitioner can choose either an accurate method with a high reduction power (such as SSMA or RMHC) or a fast method which would be able to quickly condense the training data into a smaller subset, without losing too much accuracy with respect to the original 1NN (such as HMNEI or FCNN).

If we compare these results with the ones achieved by the ANN techniques, we can state the following:

- ANN techniques remain, in general, as accurate as the 1NN classifier. Therefore, they are competitive when compared with the PS methods which are not mainly focused on removing noise from the data set, but they are outperformed by those which perform any effective competence enhancing process.
- Regarding running time, both ANN techniques show very competitive behavior when dealing with large problems. Although they did not perform any reduction at all, they are able to perform queries in a very fast way, which match the time elapsed by PS methods in the classification phase.

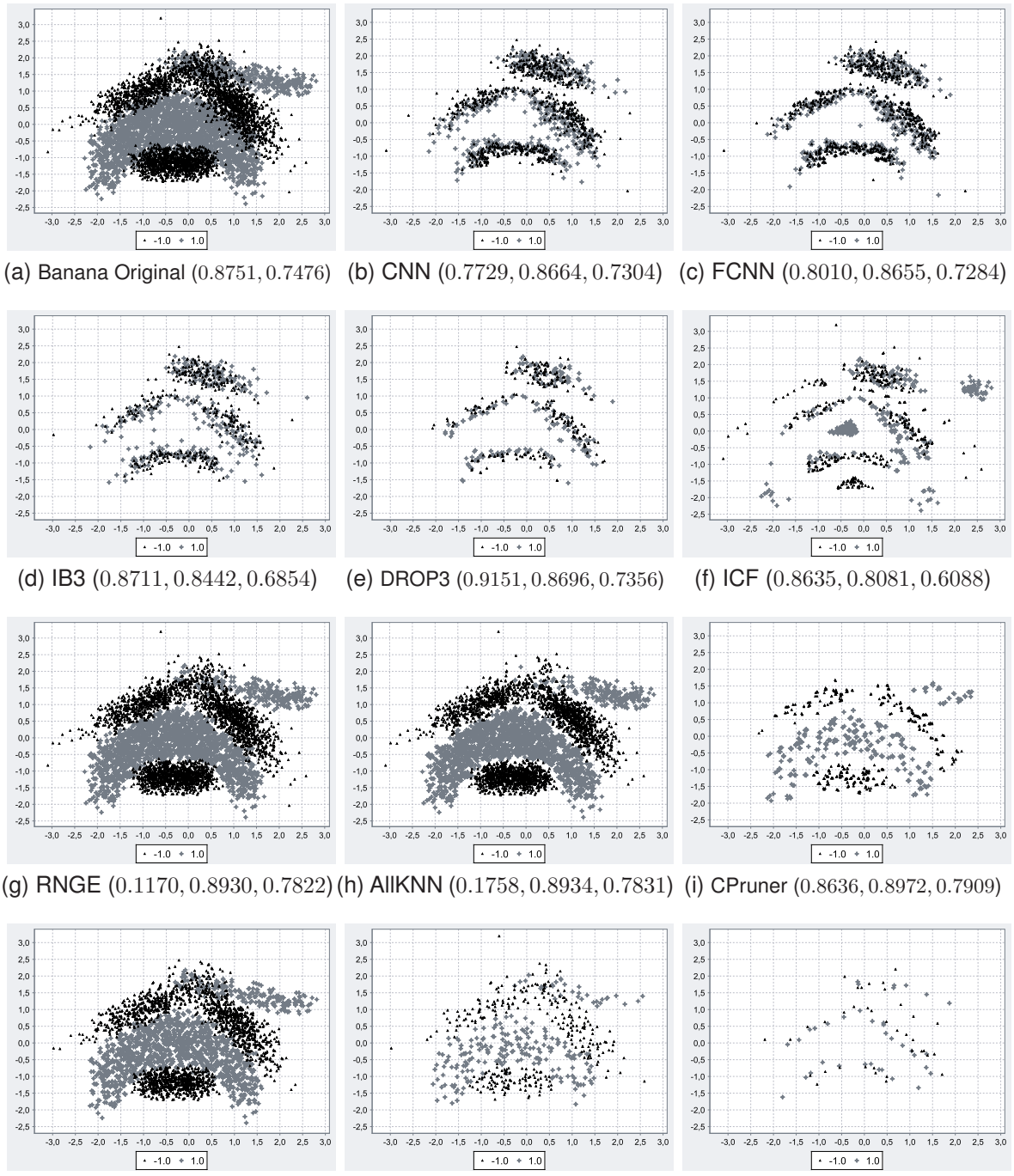
The answer to deciding whether to employ a PS or an ANN method when facing a large problem lies in the interest of the practitioner and his/her concrete objectives. For example, if the main interest is to tackle the problem quickly with a reasonable precision in classification, then an ANN method would be appropriate. On the other hand, if the user is interested in a quick method which would also be able to summarize and reduce the data to a more compact representation, without the necessity of spending additional resources on storing additional structures, a fast PS method like FCNN or HMNEI would be the best option. Finally, if the interest lies in obtaining highly precise classifiers, represented with very compact data sets, no matter how much time the PS phase takes, then a strong PS method such as SSMA or RMHC would be the best option.

## 6 Visualization of Data Subsets: A Case Study Based on the Banana Data Set

This section is devoted to illustrating the subsets selected resulting from some PS algorithms considered in our study. To do this, we focus on the *banana* data set, which contains 5,300 examples in the complete set. It is an artificial data set of 2 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 data set is illustrated in Figure 4a.

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

- Figure 4b shows the resulting subset of the classical condensation algorithm. It can be appreciated that all border points are kept but interior points are removed. The accuracy and kappa decrease with respect to the original, as is usually the case with purely condensation algorithms.
- Figure 4c illustrates the resulting subset of one of the newest condensation algorithms proposed: FCNN. The subset has a similar appearance to that obtained by CNN and the performance is also similar in both. The advantage of this method is difficult to see in graphical representations, but its improvement with respect to CNN can be seen in the experimental study section.
- Figures 4d and 4e represent the subset selected by the IB3 and DROP3 methods respectively. These methods are thought to be modifications of classical condensation algorithms but they integrate a noise filter pass, turning them into hybrid approaches. Both methods obtain a lower accuracy and kappa regarding 1NN with the original data set, but the reduction rates obtained are very high. The main factor which influences the reduction rate is noise



(a) Banana Original $(0.8751, 0.7476)$ (b) CNN $(0.7729, 0.8664, 0.7304)$ (c) FCNN $(0.8010, 0.8655, 0.7284)$

(d) IB3 $(0.8711, 0.8442, 0.6854)$ (e) DROP3 $(0.9151, 0.8696, 0.7356)$ (f) ICF $(0.8635, 0.8081, 0.6088)$

(g) RNGE $(0.1170, 0.8930, 0.7822)$ (h) AllKNN $(0.1758, 0.8934, 0.7831)$ (i) CPruner $(0.8636, 0.8972, 0.7909)$

(j) HMNEI $(0.3617, 0.8906, 0.7787)$ (k) RMHC $(0.9000, 0.8972, 0.7915)$ (l) SSMA $(0.9879, 0.8964, 0.7900)$

Fig. 4: Data Subsets in Banana Data Set

removal. Note that the difference in accuracy and kappa is higher in IB3, which suggests that IB3 penalizes the most complicated concept or class.

- Figure 4f shows the resulting subset of the ICF method. It is a curious algorithm which separates the data into smaller clusters, some isolated and others overlapped. Nevertheless, the performance achieved by this method is quite poor.
- Figures 4g and 4h depict the subset of data selected by the RNGE and AllKNN methods. Both belong to the edition approaches and the unique difference observed is that AllKNN performs a slightly more aggressive removal of instances in the decision boundaries. The performance and reduction rates are very similar between them and both improve the performance of 1NN over the original data set.
- Figures 4i and 4j represent the subset of data selected by CPruner and HMNEI methods. CPruner

performs well over this data set, obtaining good performance and reduction rates. Its way of working is based on producing isolated scatters clusters with no overlapping. On the other hand, HMNEI is one of the best methods studied in this paper. It allows one to obtain excellent behavior in terms of efficacy, closer to edition approaches, but while increasing the reduction rate.

- Figures 4k and 4l illustrate the subset of data selected by RMHC and SSMA methods. They are wrapper methods and iterate many times to obtain an optimal subset. RMHC requires as a parameter the final size of the subset selected and this parameter is very difficult to set a priori. In the *banana* case, keeping $10\%$ of prototypes may be excessive. However, SSMA can also adjust and optimize the subset to have the lowest possible number of instances. The performance in accuracy and kappa obtained improves 1NN and most of the PS methods studied with a reduction of around $98\%$. It seems that the prototypes selected are just those needed to define the 1NN regions in an accurate way.

We have seen the resulting subsets of condensation, edition and hybrid methods. The latter do not follow a specific behavior pattern, since some of them can keep the frontiers and remove noisy instances (DROP3), others can produce clusters of data (ICF) and others can identify the decision boundaries with the minimum number of prototypes (SSMA). Nevertheless, visual characteristics of selected subsets are also the subject of interest and can also help to decide the choice of a PS method.

## 7 Concluding Remarks

The present paper offers an exhaustive survey of Prototype Selection methods proposed in the literature. Basic and advanced properties, existing work and related fields have been reviewed. Based on the main characteristics studied, we have proposed a taxonomy of Prototype Selection methods. Furthermore, the most important methods have been empirically analyzed over small, medium and large sizes of classification data sets. 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:

- A researcher/practitioner interested in applying a PS method should know the characteristics needed when choosing one of them. The taxonomy proposed and the empirical study can help to make this decision.
- In the proposal of a new PS method, the best approaches and those which fit with the basic properties of the new proposal should be compared. To do this, the taxonomy and the analysis of results can guide a future proposal in the correct way.
- This paper helps non-experts in PS methods to differentiate them, to make an appropriate decision about their application and to understand their behavior.
- It is important to know the main advantages of each PS method. In this paper, many PS methods have been empirically analyzed but a specific conclusion cannot be determined on 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.
- The empirical study allows us to stress several methods among the whole set:
  - RMHC and SSMA, as representatives of the hybrid family, obtain an excellent tradeoff between reduction and classifier success.
  - RNGE achieves the highest accuracy rate within the edition family. HMNEI, belonging to hybrid methods, is also a good alternative to increase kNN efficacy.
  - As condensation methods, RNN and FCNN are the best performing techniques. FCNN is one of the fastest PS approaches.
- The PS methods in conjunction with the stratification process [31] obtain satisfactory results over large data sets. They are very competitive in comparison to approximate nearest neighbor methods (BBD and LSH). SSMA can outperform them in terms of accuracy and classification time at the expense of a high computational cost in the selection process.

We finally note that there is a web site (http://sci2s.ugr.es/pstax) associated with this paper that collects all the descriptions and implementations of the methods reviewed, as well as all detailed results obtained and statistical analysis conducted in the experimental study.

## Acknowledgments

This work was supported by TIN2008-06681-C06-01 and TIN2008-06681-C06-02.

## References

[1] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[2] I. Kononenko and M. Kukar, *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood Publishing Limited, 2007.

[3] A. N. Papadopoulos and Y. Manolopoulos, *Nearest Neighbor Search: A Database Perspective*. Springer, 2004.

[4] G. Shakhnarovich, T. Darrell, and P. Indyk, Eds., *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. The MIT Press, 2006.

[5] X. Wu and V. Kumar, Eds., *The Top Ten Algorithms in Data Mining*. Chapman & Hall/CRC Data Mining and Knowledge Discovery, 2009.

[6] D. W. Aha, Ed., *Lazy Learning*. Springer, 1997.

[7] E. K. Garcia, S. Feldman, M. R. Gupta, and S. Srivastava, "Completely lazy learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 9, pp. 1274–1285, 2010.

[8] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[9] A. K. Ghosh, "On optimum choice of k in nearest neighbor classification," *Computational Statistics & Data Analysis*, vol. 50, no. 11, pp. 3113–3123, 2006.

[10] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 607–616, 1996.

[11] Ghosh and K. Anil, "On nearest neighbor classification using adaptive choice of k," *Journal of Computational & Graphical Statistics*, vol. 16, no. 2, pp. 482–502, 2007.

[12] C. Domeniconi, J. Peng, and D. Gunopulos, "Locally adaptive metric nearest-neighbor classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1281–1285, 2002.

[13] J. Yu, J. Amores, N. Sebe, P. Radeva, and Q. Tian, "Distance learning for similarity estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 451–462, 2008.

[14] A. Argentini and E. Blanzieri, "About neighborhood counting measure metric and minimum risk metric," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 763–765, 2010.

[15] P. Cunningham, "A taxonomy of similarity mechanisms for case-based reasoning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 11, pp. 1532–1543, 2009.

[16] C.-M. Hsu and M.-S. Chen, "On the design and applicability of distance functions in high-dimensional data space," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 4, pp. 523–536, 2009.

[17] B. K. Kim and S. B. Park, "A fast k nearest neighbor finding algorithm based on the ordered partition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 761–766, 1986.

[18] S. A. Nene and S. K. Nayar, "A simple algorithm for nearest neighbor search in high dimensions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 9, pp. 989–1003, 1997.

[19] H. Samet, "K-nearest neighbor finding using maxnearestdist," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 243–252, 2008.

[20] P. Grother, G. T. Candela, and J. L. Blue, "Fast implementations of nearest-neighbor classifiers," *Pattern Recognition*, vol. 30, no. 3, pp. 459–465, 1997.

[21] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM*, vol. 45, no. 6, pp. 891–923, 1998.

[22] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Communications of the ACM*, vol. 51, no. 1, pp. 117–122, 2008.

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

[24] N. Jankowski and M. Grochowski, "Comparison of instances selection algorithms I. algorithms survey." in *ICAISC*, ser. Lecture Notes in Computer Science, vol. 3070, 2004, pp. 598–603.

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

[26] W. Lam, C. K. Keung, and D. Liu, "Discovering useful concept prototypes for classification based on filtering and abstraction." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 8, pp. 1075–1090, 2002.

[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 Recognition*, vol. 39, no. 10, pp. 1827–1838, 2006.

[28] F. Angiulli, "Fast nearest neighbor condensation for large data sets classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 11, pp. 1450–1464, 2007.

[29] J. C. Bezdek and L. I. Kuncheva, "Nearest prototype classifier designs: An experimental study," *International Journal of Intelligent Systems*, vol. 16, pp. 1445–1473, 2001.

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

[31] J.-R. Cano, F. Herrera, and M. Lozano, "Stratification for scaling up evolutionary prototype selection," *Pattern Recognition Letters*, vol. 26, no. 7, pp. 953–963, 2005.

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

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

[34] A. Cervantes, I. M. Galván, and P. Isasi, "AMPSO: a new particle swarm method for nearest neighborhood classification," *IEEE Transactions on Systems, Man and Cybernetics: Part B, Cybernetics*, vol. 39, no. 5, pp. 1082–1091, 2009.

[35] I. Triguero, S. García, and F. Herrera, "IPADE: Iterative prototype adjustment for nearest neighbor classification," *IEEE Transactions on Neural Networks*, vol. 21, no. 12, pp. 1984–1990, 2010.

[36] ——, "Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification," *Pattern Recognition*, 2011, in press. DOI:10.1016/j.patcog.2010.10.020.

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

[38] O. Luaces and A. Bahamonde, "Inflating examples to obtain rules," *International Journal of Intelligent systems*, vol. 18, pp. 1113–1143, 2003.

[39] S. García, J. Derrac, J. Luengo, C. J. Carmona, and F. Herrera, "Evolutionary selection of hyperrectangles in nested generalized exemplar learning," *Applied Soft Computing*, 2011, in press.

[40] D. B. Skalak, "Prototype and feature selection by sampling and random mutation hill climbing algorithms," in *Proceedings of the Eleventh International Conference on Machine Learning*, 1994, pp. 293–301.

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

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

[43] R. Paredes and E. Vidal, "Learning weighted metrics to minimize nearest-neighbor classification error," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1100–1110, 2006.

[44] F. Fernández and P. Isasi, "Local feature weighting in nearest prototype classification," *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 40–53, 2008.

[45] E. Alpaydin, "Voting over multiple condensed nearest neighbors," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 115–132, 1997.

[46] N. García-Pedrajas, "Constructing ensembles of classifiers by means of weighted instance selection," *IEEE Transactions on Neural Networks*, vol. 20, no. 2, pp. 258–277, 2009.

[47] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *Journal of Artificial Intelligence Research*, vol. 6, pp. 1–34, 1997.

[48] R. Paredes and E. Vidal, "Learning prototypes and distances: A prototype reduction technique based on nearest neighbor error minimization," *Pattern Recognition*, vol. 39, no. 2, pp. 180–188, 2006.

[49] C. Gagné and M. Parizeau, "Coevolution of nearest neighbor classifiers," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 21, no. 5, pp. 921–946, 2007.

[50] S.-W. Kim and B. J. Oommen, "On using prototype reduction schemes to optimize dissimilarity-based classification," *Pattern Recognition*, vol. 40, no. 11, pp. 2946–2957, 2007.

[51] A. Haro-García and N. García-Pedrajas, "A divide-and-conquer recursive approach for scaling up instance selection algorithms," *Data Mining and Knowledge Discovery*, vol. 18, no. 3, pp. 392–418, 2009.

[52] C. García-Osorio, A. de Haro-García, and N. García-Pedrajas, "Democratic instance selection: A linear complexity instance selection algorithm based on classifier ensemble concepts," *Artificial Intelligence*, vol. 174, no. 5-6, pp. 410–441, 2010.

[53] F. Angiulli and G. Folino, "Distributed nearest neighbor-based condensation of very large data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 12, pp. 1593–1606, 2007.

[54] J.-R. Cano, F. Herrera, and M. Lozano, "Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability," *Data and Knowledge Engineering*, vol. 60, no. 1, pp. 90–108, 2007.

[55] K. J. Kim, "Artificial neural networks with evolutionary instance selection for financial forecasting," *Expert Systems with Applications*, vol. 30, no. 3, pp. 519–526, 2006.

[56] J.-R. Cano, F. Herrera, M. Lozano, and S. García, "Making CN2-SD subgroup discovery algorithm scalable to large size data sets using instance selection," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1949–1965, 2008.

[57] J.-R. Cano, S. García, and F. Herrera, "Subgroup discover in large size data sets preprocessed using stratified instance selection for increasing the presence of minority classes," *Pattern Recognition Letters*, vol. 29, no. 16, pp. 2156–2164, 2008.

[58] Y. Chen, J. Bi, and J. Z. Wang, "MILES: Multiple-instance learning via embedded instance selection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 12, pp. 1931–1947, 2006.

[59] Z. Fu, A. Robles-Kelly, and J. Zhou, "MILIS: Multiple instance learning with instance selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010, in press. DOI: 10.1109/TPAMI.2010.155.

[60] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 225–252, 2008.

[61] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

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

[63] S.-W. Kim and B. J. Oommen, "On using prototype reduction schemes to enhance the computation of volume-based inter-class overlap measures," *Pattern Recognition*, vol. 42, no. 11, pp. 2695–2704, 2009.

[64] R. A. Mollineda, J. S. Sánchez, and J. M. Sotoca, "Data characterization for effective prototype selection," in *Proc. of the 2nd Iberian Conf. on Pattern Recognition and Image Analysis (ICPRIA)*, ser. Lecture Notes in Computer Science, vol. 3523, 2005, pp. 27–34.

[65] S. García, J.-R. Cano, E. Bernadó-Mansilla, and F. Herrera, "Diagnose of effective evolutionary prototype selection using an overlapping measure," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 8, pp. 1527–1548, 2009.

[66] P. E. Hart, "The condensed nearest neighbor rule," *IEEE Transactions on Information Theory*, vol. 18, pp. 515–516, 1968.

[67] G. W. Gates, "The reduced nearest neighbor rule," *IEEE Transactions on Information Theory*, vol. 22, pp. 431–433, 1972.

[68] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on System, Man and Cybernetics*, vol. 2, no. 3, pp. 408–421, 1972.

[69] J. R. Ullmann, "Automatic selection of reference data for use in a nearest-neighbor method of pattern classification," *IEEE Transactions on Information Theory*, vol. 24, pp. 541–543, 1974.

[70] G. L. Ritter, H. B. Woodruff, S. R. Lowry, and T. L. Isenhour, "An algorithm for a selective nearest neighbor decision rule," *IEEE Transactions on Information Theory*, vol. 25, pp. 665–669, 1975.

[71] I. Tomek, "An experiment with the edited nearest-neighbor rule," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 448–452, 1976.

[72] ——, "Two modifications of CNN," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 769–772, 1976.

[73] K. C. Gowda and G. Krishna, "The condensed nearest neighbor rule using the concept of mutual nearest neighborhood," *IEEE Transactions on Information Theory*, vol. 29, pp. 488–490, 1979.

[74] P. A. Devijver and J. Kittler, *Pattern Recognition, A Statistical Approach*. Prentice Hall, 1982.

[75] P. A. Devijver, "On the editing rate of the multiedit algorithm," *Pattern Recognition Letters*, vol. 4, pp. 9–12, 1986.

[76] D. Kibler and D. W. Aha, "Learning representative exemplars of concepts: An initial case study," in *Proceedings of the Fourth International Workshop on Machine Learning*, 1987, pp. 24–30.

[77] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991.

[78] B. V. Dasarathy, "Minimal consistent set (MCS) identification for optimal nearest neighbor decision system design," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 3, pp. 511–517, 1994.

[79] R. M. Cameron-Jones, "Instance selection by encoding length heuristic with random mutation hill climbing," in *Proceedings of the Eighth Australian Joint Conference on Artificial Intelligence*, 1995, pp. 99–106.

[80] C. E. Brodley, "Recursive automatic bias selection for classifier construction," *Machine Learning*, vol. 20, no. 1-2, pp. 63–94, 1995.

[81] D. G. Lowe, "Similarity metric learning for a variable-kernel classifier," *Neural Computation*, vol. 7, no. 1, pp. 72–85, 1995.

[82] J. S. Sánchez, F. Pla, and F. J. Ferri, "Prototype selection for the nearest neighbor rule through proximity graphs," *Pattern Recognition Letters*, vol. 18, pp. 507–513, 1997.

[83] U. Lipowezky, "Selection of the optimal prototype subset for 1-nn classification," *Pattern Recognition Letters*, vol. 19, no. 10, pp. 907–918, 1998.

[84] L. I. Kuncheva, "Editing for the k-nearest neighbors rule by a genetic algorithm," *Pattern Recognition Letters*, vol. 16, no. 8, pp. 809–814, 1995.

[85] L. I. Kuncheva and L. C. Jain, "Nearest neighbor classifier: simultaneous editing and feature selection," *Pattern Recognition Letters*, vol. 20, no. 11-13, pp. 1149–1156, 1999.

[86] K. Hattori and M. Takahashi, "A new edited k-nearest neighbor rule in the pattern classification problem," *Pattern Recognition*, vol. 33, no. 3, pp. 521–528, 2000.

[87] B. Sierra, E. Lazkano, I. Inza, M. Merino, P. Larrañaga, and J. Quiroga, "Prototype selection and feature subset selection by estimation of distribution algorithms. a case study in the survival of cirrhotic patients treated with TIPS," in *AIME '01: Proceedings of the 8th Conference on AI in Medicine in Europe*, ser. Lecture Notes in Computer Science, vol. 2101, 2001, pp. 20–29.

[88] V. Cerverón and F. J. Ferri, "Another move toward the minimum consistent subset: a tabu search approach to the condensed nearest neighbor rule," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 31, no. 3, pp. 408–413, 2001.

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

[90] V. S. Devi and M. N. Murty, "An incremental prototype set building technique," *Pattern Recognition*, vol. 35, no. 2, pp. 505–513, 2002.

[91] S.-Y. Ho, C.-C. Liu, and S. Liu, "Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm," *Pattern Recognition Letters*, vol. 23, no. 13, pp. 1495–1503, 2002.

[92] M. Sebban and R. Nock, "Instance pruning as an information preserving problem," in *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, 2000, pp. 855–862.

[93] M. Sebban, R. Nock, E. Brodley, and A. Danyluk, "Stopping criterion for boosting-based data reduction techniques: from binary to multiclass problems," *Journal of Machine Learning Research*, vol. 3, pp. 863–885, 2002.

[94] Y. Wu, K. G. Ianakiev, and V. Govindaraju, "Improved k-nearest neighbor classification," *Pattern Recognition*, vol. 35, no. 10, pp. 2311–2318, 2002.

[95] H. Zhang and G. Sun, "Optimal reference subset selection for nearest neighbor classification by tabu search," *Pattern Recognition*, vol. 35, no. 7, pp. 1481–1490, 2002.

[96] M. T. Lozano, J. S. Sánchez, and F. Pla, "Using the geometrical distribution of prototypes for training set condensing," in *CAEPIA*, ser. Lecture Notes in Computer Science, vol. 3040, 2003, pp. 618–627.

[97] K. P. Zhao, S. G. Zhou, J. H. Guan, and A. Y. Zhou, "C-pruner: An improved instance pruning algorithm," in *Proceeding of the 2th International Conference on Machine Learning and Cybernetics*, 2003, pp. 94–99.

[98] J.-R. Cano, F. Herrera, and M. Lozano, "Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 6, pp. 561–575, 2003.

[99] J. C. Riquelme, J. S. Aguilar-Ruiz, and M. Toro, "Finding representative patterns with ordered projections," *Pattern Recognition*, vol. 36, no. 4, pp. 1009–1018, 2003.

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

[101] F. Vázquez, J. S. Sánchez, and F. Pla, "A stochastic approach to wilson's editing algorithm," in *2nd Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*, ser. Lecture Notes in Computer Science, vol. 3523, 2005, pp. 35–42.

[102] Y. Li, Z. Hu, Y. Cai, and W. Zhang, "Support vector based prototype selection method for nearest neighbor rules," in *First International Conference on Advances in Natural Computation (ICNC)*, ser. Lecture Notes in Computer Science, vol. 3610, 2005, pp. 528–535.

[103] J. A. Olvera-López, J. F. Martínez-Trinidad, and J. A. Carrasco-Ochoa, "Edition schemes based on BSE," in *10th Iberoamerican Congress on Pattern Recognition (CIARP)*, ser. Lecture Notes in Computer Science, vol. 3773, 2005, pp. 360–367.

[104] R. Barandela, F. J. Ferri, and J. S. Sánchez, "Decision boundary preserving prototype selection for nearest neighbor classification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, no. 6, pp. 787–806, 2005.

[105] F. Chang, C.-C. Lin, and C.-J. Lu, "Adaptive prototype learning algorithms: Theoretical and experimental studies," *Journal of Machine Learning Research*, vol. 7, pp. 2125–2148, 2006.

[106] X. Z. Wang, B. Wu, Y. L. He, and X. H. Pei, "NRMCS : Noise removing based on the MCS," in *Proceedings of the Seventh International Conference on Machine Learning and Cybernetics*, 2008, pp. 89–93.

[107] R. Gil-Pita and X. Yao, "Evolving edited k-nearest neighbor classifiers," *International Journal of Neural Systems*, vol. 18, no. 6, pp. 459–467, 2008.

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

[109] E. Marchiori, "Hit miss networks with applications to instance selection," *Journal of Machine Learning Research*, vol. 9, pp. 997–1017, 2008.

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

[111] J. A. Olvera-López, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, "A new fast prototype selection method based on clustering," *Pattern Analysis and Applications*, vol. 13, no. 2, pp. 131–141, 2010.

[112] E. Marchiori, "Class conditional nearest neighbor for large margin instance selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 364–370, 2010.

[113] N. García-Pedrajas, J. A. Romero Del Castillo, and D. Ortiz-Boyer, "A cooperative coevolutionary algorithm for instance selection for instance-based learning," *Machine Learning*, vol. 78, no. 3, pp. 381–420, 2010.

[114] 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 Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[115] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[116] J. A. Cohen, "Coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, pp. 37–46, 1960.

[117] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[118] A. Ben-David, "A lot of randomness is hiding in accuracy," *Engineering Applications of Artificial Intelligence*, vol. 20, pp. 875–885, 2007.

[119] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[120] S. García and F. Herrera, "An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[121] 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," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[122] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, pp. 80–83, 1945.

**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 in 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.

**Joaquín Derrac** 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, lazy learning and evolutionary algorithms.

**José Ramón Cano** received the M.Sc. and Ph.D. degrees in computer science from the University of Granada, Granada, Spain, in 1999 and 2004, respectively.

He is currently an Associate Professor in the Department of Computer Science, University of Jaén, Jaén, Spain. His research interests include data mining, data reduction, data complexity, interpretability-accuracy trade-off, and evolutionary algorithms.

**Francisco Herrera** received the M.Sc. degree in Mathematics in 1988 and the Ph.D. degree 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 published more than 140 papers in international journals. He is coauthor of the book "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases" (World Scientific, 2001). As edited activities, he has co-edited four international books and co-edited twenty special issues in international journals on different Soft Computing topics. He acts as associated editor of the journals: IEEE Transactions on Fuzzy Systems, Mathware and Soft Computing, Advances in Fuzzy Systems, Advances in Computational Sciences and Technology, and International Journal of Applied Metaheuristic Computing. He currently serves as area editor of the Journal Soft Computing (area of genetic algorithms and genetic fuzzy systems), and he serves as member of the editorial board of the journals: Fuzzy Sets and Systems, Applied Intelligence, Knowledge and Information Systems, Information Fusion, Evolutionary Intelligence, International Journal of Hybrid Intelligent Systems, Memetic Computation, International Journal of Computational Intelligence Research, The Open Cybernetics and Systemics Journal, Recent Patents on Computer Science, Journal of Advanced Research in Fuzzy and Uncertain Systems, and International Journal of Information Technology and Intelligent and Computing.

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.