Pattern Anal Applic (2003) 6: 245–256
DOI 10.1007/s10044-003-0192-z

ORIGINAL ARTICLE

R. Barandela, J. S. Sánchez, R. M. Valdovinos

New Applications of Ensembles of Classifiers

Received: 23 July 2002 / Accepted: 1 April 2003

Abstract Combination (ensembles) of classifiers is now a well established research line. It has been observed that the predictive accuracy of a combination of independent classifiers excels that of the single best classifier. While ensembles of classifiers have been mostly employed to achieve higher recognition accuracy, this paper focuses on the use of combinations of individual classifiers for handling several problems from the practice in the machine learning, pattern recognition and data mining domains. In particular, the study presented concentrates on managing the imbalanced training sample problem, scaling up some preprocessing algorithms and filtering the training set. Here, all these situations are examined mainly in connection with the nearest neighbour classifier. Experimental results show the potential of multiple classifier systems when applied to those situations.

Keywords Algorithm scalability · Ensembles; Filtering outliers · Imbalanced training sample · Nearest neighbour rule · Preprocessing techniques

Introduction

Recently, efforts aimed at combining multiple classifiers into one classification system (ensemble of classifiers, multiple classifier systems, mixtures of experts, committees of learners, etc.) have become very popular, and are regarded as one of the most promising current research directions in machine learning and pattern recognition [1]. The main purpose for building up an ensemble is to obtain higher classification accuracy than that produced by its components (individual classifiers that make it up).

Ensembles have been defined as consisting of a set of individually trained classifiers whose decisions are combined when classifying new instances [2]. The combination can be made in many different ways. The simplest employs the majority rule in a plain voting system. Despite its simplicity, it is generally regarded as a very robust combination. More elaborate schema use weight voting rules, where each component is associated with a weight. This weight is computed while training the classifier, and must reflect how accurate the individual classifier is, as estimated by its performance on the training set. Other, more sophisticated, architectures have also been proposed, consisting of two levels of classifiers in what has been called 'stacked-generalisation' or 'meta-learning' (learning from the information generated by a set of learners [3]).

It is widely accepted that improvement in the overall predictive accuracy by the ensemble can occur only if there is diversity among its components, i.e. if the individual classifiers do not always agree. Of course, no benefit arises from combining the predictions of a set of classifiers that frequently coincide in the classifications (strongly correlated classifiers). Although measuring diversity is not straightforward [4], this classifiers' independence has been sought through different ways, by:

- Manipulating the training patterns (training each classifier on different subsets of the training prototypes): cross-validation, bagging, boosting, etc.
- Manipulating the input features (training each classifier with different subsets of the available features).
- Manipulating the class labels of the training prototypes.
- Incorporating random noise into the feature values or into some parameters of the learning model considered.

Most of the research done up to now has been concerned with the creation of ensembles consisting of classifiers based on the same learning model. Although it is likely that learning with different algorithms will produce diverse classifiers, this diversity is not guaranteed. Moreover, this approach would require the employment of an effective weighted combination, since some of these classifiers would perform much worse than others.

Ensembles based on the combination of a set of classifiers are currently used to achieve higher recognition accuracy [5]. In this paper, we explore possibilities to obtain

R. Barandela (✉) · R.M. Valdovinos
Institute Tecnológico de Toluca, Metepec, México. Email: rbarandela@hotmail.com

J.S. Sánchez
Universitat Jaume I, Castelló, Spain

other benefits from the employment of an ensemble. In particular, we present results of experiments carried out to research the convenience of using ensembles for three different tasks:

a) to cope with unbalanced training samples,
b) to get scalability of some pre-processing algorithms,
c) to filter the training sample.

In our research, we have focused on the widely used nearest neighbour rule. This selection has been motivated by the flexibility and other positive characteristics of this classification method.

The nearest neighbour rule

The Nearest Neighbour (1-NN) rule is one of the oldest and better-known algorithms for nonparametric classification [6]. The entire Training Sample (TS) is stored in computer memory. To classify a new instance, its distance is computed to each of the stored training cases. The new instance is then assigned to the class represented by its nearest neighbouring training pattern. The 1-NN rule is very popular because of its

a) conceptual simplicity,
b) easy implementation,
c) known error rate bounds, and
d) potentiality to compete favourably with other classification methods in real data applications.

From the above definition, it is obvious that the 1-NN rule suffers from two significant drawbacks: the large memory requirement needed to store the whole TS, and also the large response time needed. This disadvantage is more critical in the data mining context with huge databases [7,8]. This computational burden has been considerably reduced by the development of suitable data structures and associated search algorithms, and by proposals to reduce the TS size. On the other hand, the 1-NN rule, as any other non-parametric method, is extremely sensitive to the presence of noisy, atypical or erroneously labelled prototypes in the TS, which usually lead to a decrease in performance.

Earlier reported results stimulated research into and applications of ensembles of classification models like neural networks and decision trees, while they discouraged the use of ensembles of 1-NN classifiers. Experiments with Bagging (Bootstrap Aggregation [9]) have not shown a difference in performance in the built ensemble as compared to the single 1-NN classifier trained with the original learning set. These results have led to the conclusion that the 1-NN classifier is a very stable model: small changes in the TS do not cause serious perturbations in the behaviour of the classifier. In other words, bagging (random sub-sampling with replacement) has not proved to be effective in building ensembles from individual 1-NN classifiers with enough diversity.

Nevertheless, in the last few years, several attempts to create ensembles of 1-NN classifiers have been published. Bay [10] searched to break down the stability by constructing different nearest neighbour individual classifiers, each learning with a randomly selected subset of features. On the other hand, Skalak [11] and Alpaydin [12] based the destabilization of the 1-NN classifier on the application of different procedures for reducing the TS size. This latter approach has the additional advantage of allowing a decrease in the computational burden inherent to the 1-NN classifier. In particular, Alpaydin uses the well-known order-dependence characteristic of the Condensed Subset algorithm [13] to produce several different training samples for learning a limited number of nearest neighbour classifiers. Skalak employed a stacked-generalisation approach with a layer consisting of several nearest neighbour classifiers and a decision tree in the second level.

Kubat and Chen [14] also proposed an ensemble of several 1-NN rules, such that each independent classifier considers only one of the available features. Class assignment to new patterns is done through weighted majority voting of the individual classifiers. Their system is aimed at coping with irrelevant features, and at reducing the computational cost of the 1-NN rule.

Handling the imbalanced training sample problem

Recently, concern has arisen about the complications produced by imbalanced training samples in supervised classification models. A TS is said to be imbalanced when one of the classes (the minority one) is heavily underrepresented in comparison to the other (the majority) class. For simplicity, and following the common published practice, we consider here only two-class cases and, therefore, the examples are said to be either positive or negative (that is, either from the minority class or the majority class, respectively). This imbalanced situation is usual in several real domains where the classifier is to detect a rare but important case, such as fraudulent telephone calls, oil spills in satellite images, failures in a manufacturing process, or an infrequent disease. High imbalance is present in information retrieval tasks. It has been observed that imbalanced training samples may cause a significant deterioration in the performance attainable by supervised methods, particularly when classifying patterns belonging to the minority class.

To evaluate the performance of learning systems, a confusion matrix like that in Table 1 (for a two-class problem) is usually employed. The elements in this table characterise the classification behaviour of the given system. The columns are the actual class and the rows correspond to the predicted class. The sum of the two columns gives the

Table 1 Confusion matrix

	Actual positive	*Actual negative*
Predict positive	True Positive (TP)	False Positive (FP)
Predict negative	False Negative (FN)	True Negative (TN)

total number of prototypes in each class, which is $n^+ = TP + FN$ and $n^- = FP + TN$, respectively.

The standard evaluation measure in pattern recognition is the classification accuracy, defined as $acc = (TP + TN)/(n^+ + n^-)$. However, this form of classification accuracy assumes that the error costs (that is, the cost of a false positive or false negative) are equal. This assumption has been criticised as being unrealistic, particularly in the case of highly imbalanced datasets that generally have non-uniform error costs.

Many authors point out that the performance of a learning algorithm in applications with class imbalance should not be expressed in terms of the average classification accuracy. For instance, consider a domain where only 0.2% of patterns are positive. In such a situation, labelling all new patterns as negative would give an accuracy of 99.8%, but fails on all positive cases. Classifiers that optimise for accuracy in these problems are of questionable value, since they rarely predict the minority class. Consequently, in the presence of imbalanced datasets, it is more appropriate to use other performance measures.

Alternative criteria for evaluating classifier performance are ROC curves and the geometric mean of accuracies. These are good indicators of performance on imbalanced datasets, because they are independent of the distributions of prototypes between classes, and are thus robust in circumstances where such a distribution might change with time or be different in the training and test sets.

ROC space [15] represent the FP rate, FP/n^-, on the X-axis of a graph and the TP rate, TP/n^+, on the Y-axis. Each classifier can be represented by a point in the ROC space corresponding to its FP and TP rates. The point (0,0) corresponds to the strategy of never making a positive (minority) prediction and the point (1,1) to always predicting the positive class. The point (0,1) represents perfect classification (all positive patterns are classified correctly, and no negative case is misclassified as positive), and the line x = y represents the strategy of randomly guessing the class. In ROC analysis, a classifier A is better than a classifier B if it is located to the northwest (TP is higher, FP is lower, or both) of B in ROC space. Classifiers that admit smooth variations of several of its parameters can be represented in the ROC space by appropriate curves.

On the other hand, the geometric mean of accuracies measured separately on each class [16] is defined as $g = (acc^+ \bullet acc^-)^{\frac{1}{2}}$, where $acc^+ = TP/n^+$ is the accuracy on patterns from the positive class, and $acc^- = TN/n^-$ denotes the accuracy on patterns from the minority class. This measure closely relates with the distance to perfect classification in the ROC space.

The rationale behind this measure is to maximise the accuracy on each of the two classes while keeping these accuracies balanced. For instance, a high acc^+ by a low acc^- will result in a poor g value. The g measure has the distinctive property of being nonlinear, that is, a change in acc^+ (or acc^-) has a different effect on g depending on the magnitude of acc^+: the smaller the value of acc^+, the greater the change of g. This means that the cost of misclassifying each pattern from the minority class increases the more often positive patterns are misclassified.

In this section, the g criterion will be used to evaluate the learning algorithms, both because the interesting general properties of g, and also because the proposed classifiers do not directly have a changing parameter which properly justifies a ROC analysis.

Most of the attempts at addressing this imbalance problem can be sorted into three categories [17]:

a) Over-sampling (re-sampling some training patterns many times) the minority class so as to match the size of the other class.
b) Downsizing (under-sampling) the majority class so as to match the size of the other class.
c) Internally biasing the discriminating process so as to compensate for the class imbalance.

The two basic methods for re-sampling the TS cause the class distribution to become more balanced. Nevertheless, both strategies have shown important drawbacks. Under-sampling can throw out potentially useful data, while over-sampling increases the TS size and hence the time to design a classifier. Furthermore, since over-sampling typically replicates examples of the minority class, overfitting is more likely to occur. Recent research has focused on improving these basic methods. Kubat and Matwin [16] proposed an under-sampling procedure aimed at removing only those instances of the majority class that are 'redundant' or that 'border' the minority prototypes. They assume that these bordering negative cases are noisy examples. Chawla et al [18] combine under-sampling and over-sampling methods and, instead of over-sampling by merely replicating minority prototypes, they form new minority instances by interpolating between several examples of the minority class that lie close together.

Pazzani et al. [19] take a slightly different approach when learning from an imbalanced TS by assigning different weights to prototypes of the different classes. On the other hand, Ezawa et al [20] bias the classifier in favour of certain attributes relationships. Kubat et al [21] use some counter-examples to bias the recognition process.

Approach proposed

In this paper, we follow another procedure to cope with this imbalanced situation. Instead of using a single classifier, an ensemble is implemented. The idea is to train each of the individual components of the ensemble with a balanced learning sample. That is, to replace an individual classification model (in our case, the 1-NN rule) with an imbalanced TS, by a combination of several classifiers, each using a balanced TS for its learning process. Working in this way, it is possible to appropriately handle the difficulties of the imbalance, while avoiding the drawbacks inherent to the over- and under-sampling techniques.

To achieve this, as many training sub-samples as

Table 2 Characterisation of the experimental datasets (always two-class cases)

Dataset	Features	Training sample		Test sample		No. of classifiers
		Class 1	Class 2	Class 1	Class 2	
Phoneme	5	1268	3054	318	764	3
Satimage	36	500	4647	126	1162	11
Glass	9	24	150	5	35	7
Vehicle	18	170	508	42	126	3

required to get balanced subsets are generated. The number of sub-samples will be determined by the difference between the number of prototypes from the majority class and that of the minority class. For instance, in Glass dataset (see Table 2) the majority class is, approximately, seven times greater than the minority one. Thus, seven sub-samples are created for building up an ensemble of seven 1-NN classifiers. Each of these individual classifiers is trained with a learning set consisting of all the prototypes in the minority class and the same number of training instances selected from among those belonging to the majority class. These prototypes from the majority class that are to be included in each training sub-sample have been chosen by two different procedures. In that way, we have two different types of ensemble:

Ensemble 1: Prototypes from the majority class are randomly selected, without replacement.
Ensemble 2: Prototypes from the majority class are randomly selected, with replacement.

Experimental evaluation

The experiments here reported were conducted with four real datasets taken from the UCI Repository [22]. In each dataset, five-fold cross validation was employed. Results to be presented hereafter represent the averaged g values of the five replications. To facilitate comparison with a previously published report [16], we have employed the same four datasets and with the same changes in their structures. In the Glass set the problem was transformed to discriminate class 7 against all the other classes, and in the Vehicle dataset, the task was to classify class 1 against all the others. Satimage dataset was also mapped to configure a two-class case, the training patterns of classes 1, 2, 3, 5 and 6 were joined to form a unique class and the original class 4 was left as the minority one. Descriptions of the datasets are in Table 2, as well as the number of classifiers designed for each dataset (always in odd number, to avoid ties in voting).

Experimental results are reported in Table 3. Values of the g criterion have been increased by both ensembles in all datasets, with the only exception of Glass. The TS of this dataset suffers not only from the imbalance issue. In addition to that, the minority class is too small. The adequacy of the TS size must be measured by considering the number of training cases of the smallest class, and not that of the whole TS. For the minority class in Glass dataset, the size/dimensionality rate is very low: 2.7 examples for each attribute. Results of Ensembles 1 (selection without replacement) and 2 (with replacement) are very similar. In all the experiments, decision (class label assignment) of the ensembles was always done by simple (majority) voting. Results obtained with the method proposed by Kubat and Matwin [16] are also included in Table 3.

In Table 4, the geometric mean results of Ensemble 1 are statistically compared with those obtained with the proposal of Kubat and Matwin [16]. One-side t-tests are employed to assess the significance of the differences between the two approaches. Ensemble 1 is also compared with the performance of the single 1-NN classifier, reported as a baseline.

Ensemble 1 is significantly better than Kubat–Matwin approach in two of the employed datasets. Phoneme is the less imbalanced of the datasets: only 2.4 negative prototypes for each positive instance. Problematic characteristics of the Glass dataset have already been mentioned. In these two datasets, both techniques (Ensemble 1 and Kubat–Matwin) have not been able to improve on the results yielded with the single classifier.

Table 4 Statistical assessment of the differences between Ensemble 1 and the Kubat–Matwin proposal

Dataset	Ensemble 1 vs. Single classifier	Ensemble 1 vs. Kubat–Matwin
Phoneme	no significance	no significance
Satimage	$p < 0.001$	$p < 0.001$
Glass	no significance	no significance
Vehicle	$p < 0.001$	$p < 0.001$

Table 3 Averaged g values (and standard deviations) in the imbalanced experiments

Dataset	Single classifier	Ensemble 1	Ensemble 2	Kubat-Matwin
Phoneme	73.8 (6.0)	74.3 (8.0)	74.0 (8.0)	74.4 (7.9)
Satimage	70.9 (15.5)	79.4 (4.0)	79.6 (4.5)	71.7 (4.6)
Glass	86.7 (12.2)	86.6 (10.1)	86.0 (11.8)	86.4 (10.9)
Vehicle	55.8 (4.1)	68.4 (3.3)	68.0 (3.6)	62.0 (3.5)

Scalability of some pre-processing algorithms

Another research area that has recently seen a lot of activity in pattern recognition and data mining is the development of methods for scaling up algorithms so that they can be applied to huge databases that cannot be entirely loaded in RAM, or that make their handling prohibitively expensive.

Two possible ways to cope with the problem of having a huge input data set have been researched. One way consists of redesigning known algorithms so that, while approximately maintaining its performance, it can be applied efficiently to these large data sets [23]. The second possibility is to reduce the size of (scale down) the data set. In this second approach, one would try to obtain a reduced set such that the results produced with it would replicate almost exactly those yielded when working with all the data. Different sampling schemas have been reported to produce a sample of the data [24]. Other, more heuristic techniques, called Instance or Prototype Selection algorithms, do not rely on statistical sampling, but take advantage of peculiarities of classification algorithms. Many of these instance selection techniques aim at removing from the TS the non-representative or superfluous training patterns so as to reduce storage requirement and computational cost while maintaining or even improving the classification accuracy [25]. Two good surveys of instance selection algorithms can be found in Refs. [7,26].

Since the 1-NN rule must store all the available training patterns and search through all of them to identify a new pattern, it has large memory requirements and is slow in the classification phase. The practical importance of this subject has motivated the publication of several instance selection algorithms, many of them following the idea of Hart [13] for obtaining a consistent subset. A consistent subset of a TS is some subset that correctly classifies every pattern in such a TS using the 1-NN.

Particularly remarkable is the approach of Ritter et al [27], with a clear and precise formulation of the desired goals and of the way to reach them (the Selective Subset). The aim of the algorithm of Ritter et al is the same than in several approaches to obtain minimal consistent subsets [28–30]. Although challenging, minimality is not necessarily a good property for real problems in practice. A larger subset of prototypes can represent more accurately the original (optimal) boundaries. The convenience of using minimality (with regard to consistency) has been recently discussed [31,32].

Barandela et al. [33] presented a reduction technique (Modified Selective Subset: MSS), that rests upon a modification of the Selective algorithm. The main purpose of this modification is to strengthen the condition to be fulfilled by the resulting reduced subset to attain a better approximation to the original decision boundaries. In the algorithm of MSS, preference is given to training patterns that lie close to the original (as defined by the whole TS) 1-NN decision boundaries. This idea has already been used with different flavours to obtain consistent subsets [34,35]. MSS employs, as a criterion to measure the closeness to the boundary, the distance to its nearest enemy or Nearest Unlike Neighbour (NUN) [28]. In the work of Tomek [34], this is called NNO: nearest neighbour from the opposite class. Several experimental results with real data have shown that MSS yields higher classification accuracy than Hart's and Ritter's approaches [33]. See Appendix I for a more detailed description of the MSS technique.

On the other hand, the benefits of combining Editing [36] with the consistent subset for attaining a much more significant size reduction have often been stated [37,38]. The main purpose of the Editing technique is to remove atypical training patterns, mainly those lying in the border between overlapping classes, for increasing prediction accuracy. Atypical data are usually defined [39] as those instances that do not follow the same model as the rest of the instances in the same class. This definition "includes not only erroneous data but also surprising veridical data" [40]. Incorrectly labelled training prototypes are also regarded as atypical instances by several authors [41]. A description of Wilson's editing can be found in Appendix I.

MSS, as many other reduction techniques, and as the Editing algorithm [36], requires one to examine all the available training patterns. That is, these algorithms operate in a batch mode. This characteristic makes intractable their application when the TS size is too large. Since, paradoxically, it is on large training samples that reduction techniques are most necessary, an approach to scale them is a must. That is, an approach to scale up these algorithms is necessary to be able to scale down the sample size.

Approach proposed

Most current approaches for scaling algorithms have been looking either for effective ways for sub-sampling the training data, or for suitable data structures. Chan and Stolfo [42] employ ensembles of decision trees and parallel processors to handle large datasets. A similar approach is followed in Chawla et al. [43].

In this paper, this idea of using ensembles of classifiers is again explored, but not to scale a learning algorithm. Now, the goal is to scale the above-mentioned preprocessing techniques. Briefly, the proposed procedure consists of partitioning the TS and applying the preprocessing techniques (Edition, MSS, as well as the joint employment of both) on each sub-sample. The resulting (processed) sub-samples are then used as training sets for corresponding 1-NN classifiers. Afterwards, these individual 1-NN classifiers are used to build an Ensemble. Classification of new samples is done through employment of this ensemble of 1-NN classifiers (each with a preprocessed TS).

Experimental evaluation

Table 5 presents a description of eight datasets taken from the UCI Repository [22], and which are among those that

Table 5 Description of the datasets for the scalability experiments

Dataset	No. classes	No. features	Training set size	Test set size
Cancer	2	9	546	137
Heart	2	13	216	54
Liver	2	6	276	69
Pima	2	8	615	153
Glass	6	9	174	40
Iris	3	4	120	30
Vehicle	4	18	678	168
Wine	3	13	144	34

Table 6 Classification accuracy (and standard deviations) in the scalability experiments with a MCS

Dataset	Ensemble			
	Original	Editing	MSS	Editing + MSS
Cancer	96.9 (2.0)	96.8 (2.3)	96.5 (1.4)	95.3 (4.2)
Heart	65.2 (4.2)	67.4 (3.8)	61.5 (4.4)	67.4 (5.0)
Liver	63.8 (3.7)	69.9 (3.9)	59.1 (4.7)	66.4 (3.0)
Pima	68.9 (3.3)	71.5 (3.2)	65.1 (3.8)	71.2 (3.6)
Glass	68.0 (8.6)	59.5 (11.2)	65.0 (8.5)	61.0 (9.1)
Iris	96.0 (1.5)	97.3 (1.5)	95.9 (2.6)	97.3 (2.8)
Vehicle	64.5 (2.8)	57.6 (3.4)	63.0 (2.8)	57.8 (3.1)
Wine	74.1 (8.7)	71.8 (5.3)	67.7 (8.6)	72.4 (5.3)

have received more attention in the machine learning literature. These datasets were employed here to explore the idea of implementing an ensemble to obtain scalability of the Edition and MSS algorithms, as well as of the combination of both techniques. The experiments simulate that the whole TS cannot be entirely loaded into memory. Accordingly, the TS is divided into three subsets. Thus, the experiments reported were done with ensembles consisting of three individual components, all with the 1-NN rule and with training samples of equal sizes (one-third of the dataset). Five-fold cross validation was used in each dataset.

Results in Table 6 correspond to the classification accuracy obtained when employing simple voting in an ensemble created by partitioning the TS by a sequential selection of the patterns. The same experiments were also carried out with other schema (random selection, with and without replacement), but the classification accuracies were always lower. Results obtained when working with a single classifier (with the whole TS) are provided in Table 7 for comparison.

As can be seen from Tables 6 and 7, the classification performance of the MSS algorithm (and of this reduction technique combined with Editing) does not suffer from degradation by using them through the procedure proposed. In fact, both the resulting ensemble and single classifier obtain similar results (column 5 in both tables). This means that it is possible to scale up these methods by using an ensemble, without affecting performance.

The same conclusion can be reached from the results in Table 8. Both in the resulting ensemble and in the single classifier, similar reduction rates of the storage requirements are obtained. In fact, in six of the datasets, employment of an ensemble allows for more reduction of the TS size. On the other hand, as is to be expected, combination of Wilson's editing with the MSS results (also in the ensemble) in a rather small TS size.

As an additional and interesting point, note that the ensemble without pre-processing (i.e. with the original TS) outperforms the single classifier in five of the eight datasets (see column 2 in both Tables 6 and 7). This observation opposes the well-known statement of Breiman [9] on the stability of the nearest neighbour rule.

Table 7 Classification accuracy (and standard deviations) in the scalability experiments with a single classifier

Dataset	Single classifier			
	Original	Editing	MSS	Editing + MSS
Cancer	95.6 (2.5)	96.3 (2.3)	94.7 (2.0)	96.8 (1.9)
Heart	58.2 (6.2)	64.4 (1.6)	58.5 (7.2)	63.3 (2.4)
Liver	62.3 (4.8)	69.3 (7.0)	60.6 (6.3)	67.0 (4.2)
Pima	65.9 (5.2)	72.0 (2.9)	63.0 (4.9)	70.9 (2.3)
Glass	70.0 (5.3)	65.6 (6.7)	67.0 (4.1)	64.5 (6.9)
Iris	96.0 (1.5)	96.7 (2.4)	95.3 (5.0)	95.3 (3.0)
Vehicle	64.2 (1.8)	61.3 (2.8)	63.0 (1.7)	61.0 (2.8)
Wine	72.4 (3.6)	71.2 (9.2)	70.6 (6.2)	70.0 (8.4)

Table 8 Training sample size in the scalability experiments

Dataset	Initial size	Single classifier				Ensemble			
		Editing	MSS	Edit+MSS	%	Editing	MSS	Edit+MSS	%
Cancer	546	528.8	55.6	20.8	3.8	527.6	56.2	20.0	3.7
Heart	216	138.0	122.2	36.4	16.9	129.0	120.6	36.2	16.8
Liver	276	175.0	164.4	58.8	21.3	163.0	166.0	57.8	20.9
Pima	615	427.8	303.0	77.2	12.6	423.3	310.8	92.2	15.0
Glass	174	119.4	99.4	43.2	24.8	92.6	114.0	52.4	30.1
Iris	120	115.6	20.6	11.4	9.5	112.8	32.0	19.8	16.5
Vehicle	678	234.6	419.0	182.8	27.0	383.6	458.2	143.0	21.1
Wine	144	103.8	60.0	15.8	11.0	95.4	65.0	15.4	10.7

A possible alternative to the procedure above outlined is to partition the TS, to apply the preprocessing techniques to each sub-sample, and then to join the resulting preprocessed sub-samples to again build a (preprocessed) complete TS. Afterwards, unknown patterns are to be classified with a unique classifier (the 1-NN rule), and not with an ensemble. With this alternative, the computational requirements are reduced. However, predictive accuracy is somehow decreased (see Table 9). These results point to the convenience of the ensemble, not only to facilitate the preprocessing of the training sample, but also for the classification task.

Summarising the results reported in this section, we conclude that in applications with a very large TS (a very common situation in data mining), it is possible to divide the original TS into a set of partitions, process each sub-sample separately, and then implement an ensemble with a number of individual classifiers. This obtains similar results, in terms of classification accuracy and size reduction, to those produced when the entire TS can be loaded into memory. Therefore, ensembles can be deemed successful in scaling the editing and pruning algorithms.

Filtering the training sample

Outlier data is a concept that has been considered in statistics for some time. It has been defined as a case that differs significantly from the rest of the instances in its class or group. Difficulties for detecting these cases and to handle them have long been discussed [39]. Now the term has also come to the fore in the machine learning, pattern recognition and data mining areas. Several reports have been published about the effect of these 'noisy' or atypical patterns when included in the TS, and how to counteract this drawback.

Gopalakrishnan et al. [44] are concerned with the long training time usually required by some neural network models, and propose the use of a clustering technique for identifying noisy training cases that slow down the learning phase of these models. John [45] addresses the identification of outliers in categorical data, and promotes an iterative procedure to clean the training data. He works with the C4.5 tree induction algorithm, and removes those training instances linked to the nodes eliminated during the pruning phase (ie. those cases whose removal from the TS will most increase the model's estimate of its own performance). Afterwards, the model is again built considering only the cleaned training data. His purpose is to obtain smaller trees and with better predictive accuracy.

Within the context of the 1-NN rule, extensive efforts have also been given to improvement of the classification performance when there are outliers in the TS. For example, Wilson's original proposal [36] aims at retaining in the TS only 'good' samples (i.e. training samples that are correctly classified by the k-NN rule) (see Appendix I). Tomek [46] extended this approach with a procedure that utilised all the l-NN classifiers, with l ranging from 1 through k, for a given value of k.

Koplowitz and Brown [47] proposed an alternative to Wilson's scheme in which some samples are discarded from the TS and others are relabelled according to the classification of a k-NN rule. Devijver and Kittler [48] introduced the well-known Multiedit algorithm, which is

Table 9 Classification accuracy (and standard deviation) obtained when the TS is preprocessed with the joint application of Wilson's Editing and the MSS

Datasets	Single classifier from the start	Ensemble for preprocessing and for classification	Preprocessing separately and then joining
Cancer	96.8 (1.9)	95.3 (4.2)	96.1 (2.3)
Heart	63.3 (2.4)	67.4 (5.0)	65.9 (3.8)
Liver	67.0 (4.2)	66.4 (3.0)	65.8 (5.3)
Pima	70.9 (2.3)	71.2 (3.6)	71.8 (2.2)
Glass	64.5 (6.9)	61.0 (9.1)	64.0 (9.5)
Iris	95.3 (3.0)	97.3 (2.8)	94.0 (2.8)
Vehicle	61.0 (2.8)	57.8 (3.1)	57.5 (3.6)
Wine	70.0 (8.4)	72.4 (5.3)	68.6 (5.7)

based on iteratively using the holdout error estimate. Although the Multiedit algorithm has been proven to be asymptotically optimal, in practice it shows poor behaviour when applied to finite sets of prototypes [38].

More recently, a genetic algorithm was proposed [49] as a way of editing the 1-NN classification rule. Sánchez et al [38] introduced a method to select training prototypes by using some proximity graphs. Many other editing techniques have been further presented in the literature [50–52].

Approach proposed

Brodley and Friedl [41] presented an innovative approach to identify atypical prototypes in the TS. This approach consists of employing an ensemble of classifiers to decide which prototypes are to be considered as atypical and, consequently, removed from the TS. That is, the idea of Brodley and Friedl is to create an ensemble filter to process the TS. That filter will identify a training instance as atypical if h of the m individual classifiers cannot classify it correctly. They introduced two variants of their filtering procedure:

- Ensemble filter 1 (majority filter): an instance is considered as atypical if more than half of the m individual classifiers classify it incorrectly.
- Ensemble filter 2 (consensus filter): an instance is considered atypical if all of the m individual classifiers classify it incorrectly.

In this section, we present experimental results that include initial research on the suitability of the proposals of Brodley and Friedl, and also of a new alternative introduced in the present paper:

- Ensemble filter 3 (removals and re-labelling): an instance is removed from the TS if less than half of the m individual classifiers coincide in the label assignment. If more than half of the m individual classifiers agree, then the training prototype receives the class label assigned by the majority of the classifiers (this label can be different from the original one).

Experimental evaluation

The three procedures mentioned have been evaluated with the same datasets as in the preceding section (see Table 5), with the exception of the Glass dataset, because of the very small size in one of its classes. Five-fold cross validation was also employed.

Table 10 reports the results in averaged accuracy rates and standard deviations. These results were obtained when classifying the corresponding test sets using the 1-NN rule, with the original (unfiltered) TS and with the TS filtered by the three ensembles. The filtering ensembles were built with four individual classifiers from the machine learning and statistical pattern recognition areas: 1-NN, Fisher's Linear Discriminant Analysis (LDA) [53], a Decision Tree (C4.5) and a Multi-Layer Perceptron neural model (for each dataset, the number of hidden nodes – only one hidden layer – was set as the number of features plus one; the stopping criteria for the learning phase were: 30,000 as the maximum number of iterations in the training process, or a system error less than 0.0001).

It is worth mentioning that the proposed alternative (Ensemble filter 3) works in a manner similar to the Generalised Editing technique [47], combining elimination of some prototypes and label change of some other training instances. In general, it can be seen that filtering the TS with the help of an ensemble increases the classification accuracy. This improvement is more evident when preprocessing of the TS is done with our filtering ensemble: it outperforms the other ensembles in four of the seven databases.

Similar results are obtained when the filtered TS (with the ensemble procedures) is used for learning of the Multi-Layer Perceptron model. These results are shown in Table 11. Filtering Ensemble 3 produces an improvement in the accuracy of this neural network model, except for Vehicle and Wine. As in Table 10, filtering Ensemble 2 is the less effective of the three procedures.

The performance of the decision tree model (C4.5) employed also improves when the TS is filtered by the experimented procedures. Here, however, the increase in classification accuracy is not so important. In the case of the fourth individual component of the filtering ensembles, LDA, pre-processing of the TS does not yield good results – see Tables 12 and 13, respectively. It is important to note that LDS is a parametric classification method and, therefore, it is less sensitive to the presence of atypical or noisy training patterns.

Concluding comments

In this paper, the use of ensembles of classifiers has not directly focused on improving the classification accuracy. Instead, ensembles have been considered to manage several problems in applications in the machine learning, data mining and pattern recognition areas: the imbalanced TS problem, the scalability of some algorithms, and the filtering of the training set. From the experiments presented, it can be concluded that, in general, the use of an ensemble of classifiers constitutes a suitable alternative to facing these practical problems.

In the imbalanced TS problem, it has been shown that it is possible to increase the geometrical mean (the measure of performance) as the result of the balance obtained in each individual component of the ensemble. In this manner, the negative effects of the imbalance can be avoided without replicating the minority class (that does not add new information to the system, and produces an increment in the computational burden of learning algorithms such as the 1-NN rule or the Multi-Layer Perceptron), and without discarding prototypes from the majority class (which can result in throwing away some

Table 10 Accuracy rate (and standard deviation) with the Nearest Neighbour classifier in the filtering experiments

Dataset		After the filtering Ensemble No.		
	Not filtered	1	2	3
Cancer	95.6 (2.5)	96.2 (2.4)	95.9 (2.8)	96.0 (2.2)
Heart	58.2 (6.2)	61.5 (4.6)	48.2 (15.5)	62.0 (2.8)
Iris	96.0 (1.5)	96.0 (1.5)	96.0 (1.5)	97.3 (2.8)
Liver	62.3 (4.8)	65.5 (4.7)	63.5 (4.9)	64.9 (3.5)
Pima	65.9 (5.2)	71.7 (3.9)	71.0 (7.4)	73.6 (4.4)
Vehicle	64.2 (1.8)	63.9 (3.5)	64.6 (2.6)	65.4 (2.8)
Wine	72.4 (3.6)	72.1 (3.4)	72.1 (3.4)	70.0 (4.8)

Table 11 Accuracy rate (and standard deviation) with the Multi-Layer Perceptron in the filtering experiments

Dataset		After the filtering Ensemble No.		
	Not filtered	1	2	3
Cancer	91.7(6.9)	95.6(1.9)	93.7(2.2)	95.6(1.9)
Heart	69.3(5.8)	76.7(3.8)	70.7(7.8)	80.7(5.0)
Iris	94.0(6.4)	96.0(5.5)	95.3(5.5)	97.3(2.8)
Liver	66.1(3.0)	65.4(5.5)	61.4(4.2)	67.5(2.4)
Pima	73.9(3.5)	75.7(4.9)	70.1(8.0)	74.8(3.1)
Vehicle	93.0(5.2)	30.6(8.3)	32.7(10.3)	69.6(4.5)
Wine	98.2(4.1)	97.1(2.9)	97.1(2.9)	92.4(3.4)

Table 12 Accuracy rate (and standard deviation) with the C4.5 decision tree in the filtering experiments

Dataset		After the filtering Ensemble No.		
	Not filtered	1	2	3
Cancer	92.4(4.2)	92.4(4.2)	92.4(4.2)	92.4(7.2)
Heart	77.4(7.0)	78.9(4.5)	66.7(19.6)	78.5(5.0)
Iris	94.7(3.0)	96.0(1.5)	94.7(3.0)	96.0(1.5)
Liver	61.4(6.8)	61.4(2.4)	65.8(1.3)	63.5(4.7)
Pima	72.8(5.2)	75.6(3.4)	75.8(2.6)	73.3(7.4)
Vehicle	72.9(3.7)	72.6(2.9)	71.4(2.9)	71.5(1.5)
Wine	90.0(2.6)	90.6(3.2)	90.0(2.6)	88.8(7.3)

Table 13 Accuracy rate (and standard deviation) with the Linear Discriminant Analysis in the filtering experiments

Dataset		After the filtering Ensemble No.		
	Not filtered	1	2	3
Cancer	96.0(3.1)	95.9(3.4)	96.0(3.1)	95.7(3.4)
Heart	83.3(4.7)	83.3(3.2)	83.7(4.0)	82.2(3.8)
Iris	97.3(1.5)	98.0(1.8)	98.0(1.8)	97.3(2.8)
Liver	64.3(4.5)	64.3(3.3)	64.9(3.9)	63.2(4.7)
Pima	75.8(4.4)	75.6(4.1)	75.7(4.0)	74.5(4.6)
Vehicle	78.3(2.4)	77.8(1.9)	78.4(2.1)	76.2(2.6)
Wine	98.2(2.6)	98.2(2.6)	98.8(2.6)	96.5(3.8)

useful information). We intend to do research to try to determine the contribution of the ensemble (besides the balance obtained in each sub-sample) to the increase in the g value.

For scalability of the editing and pruning algorithms (and of the combination of both), it is important to note that, in this case, classification accuracy improvement was not the goal. A multiple classifier system has been shown as an easy and adequate solution to obtain scalability of these preprocessing techniques without performance deterioration, in terms of both classification accuracy and sample size reduction.

Employment of an ensemble for filtering the training sample (and, consequently, to improve accuracy) is also

a promising research line. The results in Table 10 for filtering ensemble 3, that we introduced in the paper, compare with those obtained using Wilson's Editing (see column 2 in Table 7), a widely accepted technique for preprocessing the training sample. Now we intend to explore the convenience of using this filtering ensemble repeatedly, and combined with ensemble filtering 1, in a manner similar to the Depuration methodology [50].

The results reported should be viewed as a first step towards new and unconventional applications of ensembles of classifiers and, therefore, there is a number of extensions and improvements to the ideas introduced. In particular, it is important to consider other procedures to select the individual sub-samples. It is interesting to observe that, in the experiments reported in Sects. 3.2 and 4.2, random selection with replacement (as in bagging) has not offered good results. Previous reports of ensembles with 1-NN classifiers (see Sect. 2) have also not employed bagging. In addition, more advanced methods to combine the decisions of the components of an ensemble (not just the simple voting schema reported in the present paper) must be researched.

We are, at present, conducting experiments to cope with the imbalance issue in multi-class problems. On the other hand, in the case of scalability, we are doing some research to work with multiple classifier systems when manipulating both prototypes and attributes.

Acknowledgements

This work has been partially supported by grants 32016-A from the Mexican CONACYT, SAB2001-0004 from the Spanish MECD and TIC2000-1703-C03-03 from the Spanish CICYT. The three anonymous reviewers also helped to clarify the work, as well as pointing out areas for improvement.

APPENDIX I: The Modified Selective Subset (MSS)

As already stated in Sect. 4, MSS is an instance selection technique that rests upon a modification of the Selective Subset. In MSS, the first two definitions of the Selective Subset are kept as in the work of Ritter et al:

a) A prototype x_i is a related neighbour of another prototype x_j, both from the same class, if x_i is closer to x_j than the nearest enemy of x_j.
b) The set of all related neighbours of x_i is presented by Y_i.

But definition (c) is changed in the following way:

c′) The Modified Selective Subset (MSS) is defined as that subset of TS which contains, for every x_i in TS, that element of its Y_i that is the nearest to a class other than that of x_i

The main purpose of this modification is to strengthen the condition to be fulfilled by the reduced subset so as to attain a best approximation to the decision boundaries. To provide better subsets of prototypes — and also to give an efficient alternative to the Selective algorithm of Ritter et al — a greedy algorithm is introduced to obtain selective prototypes in such a way that preference is given to training patterns that lie close to the original (as defined by the whole TS) 1-NN decision boundaries. The criterion used here to measure the closeness to the boundary is the distance to its *nearest enemy* or Nearest Unlike Neighbour (NUN). The nearest enemy of x_i is the training pattern x_k that has been found to be the nearest neighbour of x_i when considering only those training patterns from all classes other than that of x_i. Using this measure, it is possible to define the best selective subset as the one that contains the best-related neighbour for each prototype in the TS. In this context, best means lower distance to its nearest enemy.

The conceptual simplicity of the MSS algorithm contrasts with the algorithm proposed by Ritter et al. In fact, the implementation is also much more straightforward. There is no need to compute related neighbours or to maintain any matrix in memory. A possible efficient implementation of this algorithm makes use of a sorting algorithm followed by two nested *for* loops, as shown in Fig. 1.

C denotes a set of prototypes that still have to fulfil the selective property (these prototypes do not yet have a related neighbour in MSS). D_j refers to the distance from x_j to its nearest enemy.

Once the prototypes have been sorted, a unique pass through the training set suffices to select the above-defined best selective subset. This requires a quadratic number of basic operations like checking set membership, distance calculation or retrieval, and set updates. Altogether, the proposed method constitutes a polinomic way of obtaining a selective subset that is not minimal, but usually gives better classification accuracy.

Wilson's editing technique

The idea of the procedure of Wilson consists of discarding prototypes that significantly deviate from the general tend-

```
Let C = TS
MSS starts as an empty set
Sort the prototypes in C according to increasing values of Di.

   For  i = 1, ..., n  do
     Put add = FALSE
     For j = 1, ..., n  do
       if (xi is in C) AND (d(xi , xj) < Dj) then
         Remove xj from C
         Put add = TRUE
     endfor
   if  add  then put xi in MSS
   if (C has resulted emptied)  then return MSS
   endfor
```

Fig. 1 Efficient implementation of the proposed modified selective algorithm

ency in their class and also prototypes lying in the overlapping zones between classes. In this way, the processed TS is able to classify new patterns in a close-to-optimal way. Apart from its optimal behaviour, editing prototypes should be almost compulsory in situations in which an optimal (or at least fair) labelling of the training is impossible, because it relies on a very difficult or costly (usually manual) task. By removing from the TS those training patterns that do not coincide with the majority of their k nearest neighbours, the Editing technique eliminates noisy as well as close border instances, leaving smoother decision boundaries. The algorithm has the following steps:

1. For every x_i in TS, find the k ($k = 3$ has been recommended) nearest neighbours of x_i among the other prototypes, and the class associated with the larger number of patterns among these k nearest neighbours. Ties would be randomly broken whenever they occur.
2. Edit the TS by deleting those training patterns x_i, whose identification label does not agree with the class associated with the largest number of k nearest neighbours, as determined previously.

References

1 Dietterich TG (1997) Machine learning research: four current directions. AI Mag 68:97–136
2 Opitz D, Maclin R (1999) Popular ensemble methods: an empirical study. J Artif Intell Res 11:169–198
3 Fan DW, Chan PK, Stolfo SJ (1996) A comparative evaluation of combiner and stacked generalization. Proceedings AAAI Workshop on Integrating Multiple Learned Models 40–46
4 Kuncheva LI, Whitaker CJ (2003) Measures of diversity in classifier ensembles. Machine Learn 51:181–207
5 Giacinto G, Roli F (2001) Dynamic classifier selection based on multiple classifier behavior. Patt Recogn 2001:1879–1881
6 Dasarathy BV (1991) Nearest Neighbor Norms: NN Pattern Classification Techniques. IEEE Computer Society Press, Los Alamitos, CA
7 Brighton H, Mellish C (2002) Advances in instance selection for instance-based learning algorithms. Data Mining Knowl Discovery 6:153–172
8 Reinartz T (2002) A unifying view on instance selection. Data Mining Knowl Discovery 6:191–210
9 Breiman L (1996) Bagging predictors. Machine Learn 24: 123–140
10 Bay S (1998) Combining nearest neighbor classifiers through multiple feature subsets. Proceedings 15th International Conference on Machine Learning 37–45
11 Skalak DB (1996) Prototype Selection for Composite Nearest Neighbor Classification. PhD Thesis, University of Massachusetts
12 Alpaydin E (1997) Voting over multiple condensed nearest neighbors. Artif Intell Res 11:115–132
13 Hart P (1968) The condensed nearest neighbor rule. IEEE Trans Info Theory IT-14:505–516
14 Kubat M, Chen K (1998) Weighted projection in nearest-neighbor classifiers. Proceedings First Southern Symposium on Computing, University of Southern Mississippi
15 Swets J, Dawes R, Monahan J (2000) Better decisions through science. Sci Am 82–87
16 Kubat M, Matwin S (1997) Addressing the curse of imbalanced training sets: one-sided selection. Proceedings 14th International Conference on Machine Learning 179–186
17 Eavis T, Japkowicz N (2002) A recognition-based alternative to discrimination-based multi-layer perceptrons. Advances in Artificial Intelligence, Lecture Notes in Computer Science 1822:280–292
18 Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2000) SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
19 Pazzani M, Merz C, Murphy P, Ali K, Hume T, Brunk C (1994) Reducing mis-classification cost. Proceedings 11th International Conference on Machine Learning, San Francisco, CA 217–225
20 Ezawa KJ, Singh M, Norton SW (1996) Learning goal oriented Bayesian networks for telecommunications management. Proceedings 13th International Conference on Machine Learning, Bari, Italy 139–147
21 Kubat M, Holte R, Matwin S (1998) Detection of oil-spills in radar images of sea surface. Machine Learn 30:195–215
22 Merz CJ, Murphy PM (1998) UCI Repository of machine learning databases. University of California, Irvine. http://www.ics.uci.edu/~mlearn
23 Provost F, Kolluri V (1999) A survey of methods for scaling up inductive algorithms. J Data Mining Knowledge Discovery 3:131–169
24 Chauchat J, Boussaid O, Amoura L (1998) Optimization sampling in a large database for induction trees. Proceedings JCIS'98-Association for Intelligent Machinery 28–31
25 Lam W, Keung D, Liu D (2002) Discovering useful concept prototypes for classification based on filtering and abstraction. IEEE Trans Patt Anal Machine Intell 24(8):1075–1090
26 Wilson DR, Martinez TR (2000) Reduction techniques for instance-based lerning algorithms. Machine Learn 38(3): 257–286
27 Ritter GL, Woodritz HB, Lowry SR, Isenhour TL (1975) An algorithm for selective nearest neighbor rule. IEEE Trans Infor Theory IT-21: 665–669
28 Dasarathy BV (1994) Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design. IEEE Trans Syst, Man Cybern 24(3): 511–517
29 Kuncheva LI, Bezdek JC (1998) Nearest prototype classification: Clustering, genetic algorithms, or random search? IEEE Trans Syst, Man Cybern 28(1):160–164
30 Cerverón V, Ferri FJ (2001) Another move toward the minimum consistent subset: A tabu search approach to the condensed nearest neighbor rule. IEEE Trans Syst, Man Cybern, Part B 31(3):408–413
31 Zhang H, Sun G (2002) Optimal reference subset selection for nearest neighbor classification by tabu search. Patt Recogn 35(7):1481–1490
32 Devi VS, Murty MN (2002) An incremental prototype set building technique. Patt Recogn 35(2):505–513
33 Barandela R, Cortés N, Palacios A (2001) The nearest neighbor rule and the reduction of the training sample size. Proceedings 9th Spanish Symposium on Pattern Recognition and Image Analysis I:103–108
34 Tomek I (1976) Two modifications of CNN. IEEE Trans Syst, Man Cybern 7(2):769–772
35 Chidananda-Gowda K, Krishna G (1979) The condensed nearest neighbor rule using the concept of mutual nearest neighborhood. IEEE Trans Infor Theory 25(4):488–490
36 Wilson DL (1972) Asymptotic properties of Nearest Neighbor rules using edited data sets. IEEE Trans Syst, Man Cybern SMC-2:408–421
37 Barandela R (1987) The NN rule: an empirical study of its methodological aspects Doctoral Thesis, Institute of Cybernetics, Berlin
38 Sánchez JS, Pla F, Ferri F (1997) Prototype selection for the nearest neighbor rule through proximity graphs. Patt Recogn Lett 18(6):507–513
39 Weisberg S (1985) Applied Linear Regression. Wiley
40 John GH (1995) Robust decision trees: Removing outliers from data. Proceedings First International Conference on Knowledge Discovery and Data Mining, AAAI Press 174–179
41 Brodley CE, Friedl MA (1999) Identifying mislabeled training data. J Artif Intell Res 11:131–167
42 Chan PK, Stolfo SJ (1995) Learning arbiter and combiner trees from partitioned data for scaling machine learning. Proceedings First International Conference on Machine Learning and Data Mining 39–44
43 Chawla N, Eschrich S, Hall LO (2001) Creating Ensembles of Classifiers. IEEE International Conference on Data Mining 580–581
44 Gopalakrishnan M, Sridhar V, Krishnamurthy H (1995) Some applications of clustering in the design of neural. Patt Recogn Lett 16:59–65
45 John GH (1997) Enhancements to the Data Mining Process. PhD Thesis, Stanford University
46 Tomek I (1976) An experiment with the edited nearest neighbor rule. IEEE Trans Syst, Man Cybern SMC-6:448–452
47 Koplowitz J, Brown TA (1981) On the relation of performance to editing in nearest neighbor rules. Patt Recogn 13:251–255
48 Devijver PA, Kittler J (1982) Pattern Recognition: A Statistical Approach. Prentice Hall, London

49 Kuncheva LI (1995) Editing for the k-nearest neighbors rule by a genetic algorithm. Patt Recogn Lett 16:809–814
50 Barandela R, Gasca E (2000) Decontamination of training samples for supervised pattern recognition methods. Advances in Pattern Recognition, Lecture Notes in Computer Science 1876:621–630
51 Sánchez JS, Barandela R, Marqués AI, Alejo R, Badenas J (2003) Analysis of new techniques to obtain quality training sets. Patt Recogn Lett 24(7):1015–1022
52 Sánchez JS, Barandela R, Ferri FJ (2002) On filtering the training prototypes for nearest neighbour classification. In: M.T. Escrig et al (eds) Topics in Artificial Intelligence, Lecture Notes in Artificial Intelligence, Springer-Verlag 2504:239–248
53 Hand DJ (1997) Construction and Assessment of Classification Rules. Wiley, Chichester

Ricardo Barandela received the PhD in the Institute for Cybernetics, Berlin, in 1987, and has held a number of positions in academia since then. He is currently a Senior Researcher at the Instituto Tecnológico de Toluca, México. He has authored more than 50 scientific publications. His research interests are pattern recognition and data mining; specifically, the development of learning techniques for supervised methods, the cleaning of the training set, and instance selection techniques.

Dr. J.S. Sánchez is an Associate Professor at the Department of Program Languages and Information Systems of Jaume I University (Castellón, Spain). He received a BSc in computer science from Technical University of Valencia in 1990 and a PhD in computer science Engineering from Jaume I University in 1998. He is author or co-author of more than 40 scientific publications and co-editor of two books. He has served as guest editor for the special issue on 'Advances in Pattern Recognition and Image Analysis' of the International Journal of Pattern Recognition and Artificial Intelligence, and for the special issue on 'Pattern Recognition and Image Analysis in Cybernetic Applications' in Cybernetics & Systems. His current research interests lie in the area of pattern recognition, including classification, feature and prototype selection, decision tree induction and also application of different computational geometry tools to a number of pattern recognition problems.

Rosa María Valdovinos is currently a PhD student at the Instituto Tecnológico de Toluca, México. Her research interests include multiple classification systems and medical applications of pattern recognition.

Originality and contribution

This paper focuses on using ensembles of classifiers for handling some important problems present in different pattern recognition domains. More specifically, a combination of classifiers is proposed to manage the class imbalance problem, the scalability of some learning algorithms, and the filtering of the training sample. These problems are common in many current applications, such as data mining, remote sensing, text categorisation and retrieval of multimedia databases.

One of the main contributions of this paper is the experimental study of various ensembles of classifiers to show their applicability to the above-mentioned practical problems. The results presented here demonstrate that a combination of classifiers can be a useful tool to successfully tackle those situations. This paper can stimulate employment of ensembles of classifiers not only as a way of increasing classification accuracy, but also as a tool to undertake important tasks in pattern recognition.