ARTÍCULO

On the Use of Labelled and Unlabelled Data to Improve Nearest Neighbor Classification

F. Vázquez

Dept. de Computación, Universidad de Oriente
Av. Patricio Lumumba s/n, 90100 Santiago de Cuba, Cuba
fvazquez@csd.uo.edu.cu

J.S. Sánchez, F. Pla

Dept. de Llenguatges i Sistemes Informàtics, Universitat Jaume I
Av. Sos Baynat s/n, 12071 Castelló de la Plana, Spain
{sanchez,pla}@uji.es

Abstract

The present paper shows that the accuracy of nearest neighbor classifiers can be improved by incorporating large amounts of unlabelled patterns into a training set with a (possibly) reduced number of labelled instances. The semi-supervised learning algorithm here introduced is primarily based on a set of techniques strongly related with the popular nearest neighbor classifier, mainly in the direction of filtering the training set. Experimental results, obtained using several benchmark data sets taken from the UCI Machine Learning Database Repository, show that the employment of unlabelled data can effectively reduce classification error by up to 16%. In order to achieve such an increase in performance, it is necessary to conveniently process the unlabelled patterns by means of some editing (filtering) technique. Otherwise, errors produced by misclassifications could be incorporated into the training set, thus importantly degrading the final classification accuracy.

Key words: Semi-supervised Learning, Editing, Classification, Nearest Neighbor Rule, Nearest Centroid Neighborhood.

1 Introduction

Learning algorithms have been traditionally sorted into two broad categories: supervised and unsupervised, depending on whether labelled data are available or not. In a supervised scenario, the learner is mainly based on the information supplied by a a set of labelled instances (training set, TS) that are assumed to correctly represent all the relevant classes. Violation of this assumption may seriously deteriorate the final classification accuracy achieved by the learning system.

Supervised classification methods usually operate in two steps: a) the *learning or training phase*, for the system to acquire the necessary knowledge from the labelled instances to make itself able to differentiate among the regarded classes; and b) the *classification or operational phase*, wherein the system proceeds to identify new unknown cases as members of the considered classes. Second stage is not started before completion of the

first one and thereafter, no new knowledge is attained.

In the unsupervised learning problem, the learner is provided with only unlabelled examples. The task is to find "clusters" or groups of similar cases that probably correspond to the underlying classes. Unsupervised learning is often applied to discover structure, regularities or categories in the data, but typically requires human analysis to determine whether the discovered regularities are interesting, and to determine the true correspondence between clusters and meaningful categories.

Since the early 90's a third approach to learning, namely *partially supervised*, has received much attention [5, 6, 7, 22, 23, 27]. This paradigm conceptually represents a compromise between supervised and unsupervised learning, thus using a (generally) small number of labelled instances together with a (possibly) large set of unlabelled samples. Relevance of partially supervised learning systems is due to the fact that in many practical applications, collecting labelled training instances can be costly and time-consuming, while it is frequently easy to obtain unlabelled examples. Consequently, it results interesting to develop algorithms capable of employing both labelled and unlabelled data for classification. Learning from partially labelled data is also referred to as *semi-supervised learning* [3, 20].

This paper presents an idea to implement a classification system that not only can learn by operating with the labelled training instances, but could also benefit from the experience obtained when classifying new unlabelled patterns. The approach for working with an "ongoing learning" capability presents some interesting advantages: the classifier is more robust because errors or omissions in the original TS can be further corrected during operation and, on the other hand, the system is capable to continue adapting itself to a possibly changing (non-stationary) environment.

The ultimate aim is to facilitate the learning system to progressively increase its knowledge and consequently, to enhance the final classification accuracy. In our proposal, the nearest neighbor (NN) rule is employed as the central classifier, mainly because of its flexibility. Because a basic goal is to make the ongoing learning procedure as automatic as possible, it has been designed to work by incorporating new examples into the TS after they have been labelled by the own system. This way, however, presents the danger of performance deterioration by the inclusion of potentially mislabelled patterns to the TS. In order to minimize the risk of introducing these errors, we will employ some filters that detect and discard those possibly mislabelled cases.

The rest of the paper is organized as follows. Section 2 provides a general description of the k-NN rule along with the most important pros and cons of using this classifier in real-world domains. Moreover, Section 2 also reviews several editing procedures and describes a recent filtering algorithm based on an estimate of class conditional probabilities. In Section 3, we introduce the new learning system proposed in the present paper. Section 4 describes the experimental set-up and the data sets used in the corresponding empirical study. Next, in Section 5 we discuss the results. Finally, the main conclusions of this work and possible directions for future research are outlined in Section 6.

2 The k-Nearest Neighbors Rule

One of the most widely studied supervised classification approaches corresponds to the well-known k-NN decision rule [10]. In brief, given a set of n previously labelled examples, say $X = \{(x_1, \omega_1), (x_2, \omega_2), \ldots, (x_n, \omega_n)\}$, the k-NN classifier consists of assigning a new input sample x to the class most frequently represented among the k closest instances in the TS, according to a certain similarity measure (generally, the Euclidean distance metric). A particular case of this rule is when $k = 1$, in which an input sample is decided to belong to the class indicated by its closest neighbor.

Several properties make the k-NN classifier quite attractive, including the fact that the asymptotic risk (i.e., when $n \to \infty$) tends to the optimal Bayes risk as $k \to \infty$ and $k/n \to 0$ [9]. Moreover, if $k = 1$, the upper bound of the 1-NN classification error rate is approximately twice the Bayes error [10]. The optimal behavior of this rule in asymptotic classification performance along with a conceptual and implementational simplicity make it a powerful classification technique capable of dealing with arbitrarily complex problems, provided that there is a large enough number of training instances available.

However, in many practical situations, such a theoretical maximum can hardly be achieved due to certain inherent weaknesses that significantly reduce the effective applicability of k-NN classifiers. In particular, the performance of these rules, as with most non-parametric classification approach, is extremely sensitive to data complexity [11].

For example, classification accuracy of k-NN classifiers significantly drops down in domains where many data attributes are irrelevant [24]. Such attributes inappropriately affect the values returned by most dissimilarity metrics. Another problem using the k-NN rule refers to the seeming necessity of a lot of memory and computational resources (especially, in applications with a huge number of training examples). On the other hand, these classifiers cannot be straightforwardly employed in domains with missing attributes. Also, the class imbalance (i.e., high differences in class distributions) has been reported as an obstacle on applying distance-based algorithms to real-world problems [16].

On the other hand, class overlapping and noise or imperfections in the TS negatively affect the performance of the k-NN classifiers, and this has been widely demonstrated in many empirical studies (e.g., see [26]). That is the reason why a considerable amount of works have been devoted to improve the classification accuracy by eliminating outliers from the original TS and also cleaning possible overlapping between classes. This strategy has generally been referred to as *editing* or *filtering* [13].

2.1 Some Editing Algorithms

The general idea behind almost any editing procedure consists of estimating the true classification of instances in the TS to retain only those which are correctly labelled. Differences among most editing schemes refer to the classification rule employed for editing purposes along with the error estimate and the stopping criterion [14].

The first proposal to select a representative subset of labelled instances corresponds to Wilson's editing [30], in which the k-NN classifier is appropriately used to keep in the TS only good examples (that is, training instances that result correctly classified by the k-NN rule). Tomek [28] extended this scheme with a procedure that utilized all the l-NN classifiers, with l ranging from 1 through k, for a given value of k.

The generalized editing [19] consists of removing some "suspicious" instances from the TS and also changing the class labels of some other instances. Its purpose is to cope with all types of imperfections of the training instances (mislabelled, noisy and atypical cases). Recently, the generalized editing and Wilson's algorithm have been jointly used for the depuration method [2].

In the case of editing algorithms based on the leaving-one-out error estimate (Wilson's scheme and its relatives), the statistical independence between control and training instances cannot be assumed because their functions are interchanged. In order to achieve this statistical independence, classification of instances can be performed in a holdout manner. Thus, the Holdout editing [13] consists of randomly partitioning the initial TS into $b > 2$ blocks of instances, $B_1, ..., B_b$, and then eliminating cases from each block using only two independent blocks at the same time. Devijver and Kittler [13] also introduced the Multiedit algorithm, which basically corresponds to an iterative version of the Holdout scheme using the 1-NN rule.

A genetic algorithm [21] was also applied to define an edited set for the NN rule. Two different criteria were employed as the fitness function: the apparent error rate, and a criterion based on the certainty of the classification. The empirical results showed that the latter selected a population with chromosomes corresponding to subsets of the initial TS that provide higher classification accuracy in comparison with the whole original set, with random selection and with Wilson's technique.

Sánchez et al. [25] presented an editing algorithm based on some types of proximity graphs, such as the Gabriel Graph and the Relative Neighborhood Graph. This firstly computes the corresponding graph structure and then eliminates instances incorrectly classified by its graph neighbors. On the other hand, a combined editing-condensing scheme was also introduced to remove internal instances as well as border cases by using the concept of graph neighbors.

The rationale of the edited k-NN rule proposed by Hattori and Takahashi [15] is very similar to that of Wilson's scheme. In this method, the condition for an instance p to be included in the edited set is that all the k nearest neighbors must be from the class to which p belongs. Accordingly, this con-

dition is much more severe than that in Wilson's algorithm and as a consequence, the number of instances in the resulting edited set is equal to or less than that in Wilson's edited set.

The ACC filtering technique [17] tries to find center instances of compact regions by considering the classification performance of each example in the TS. Each training instance is classified by its nearest neighbor. If it is correctly classified, then accuracy of its nearest neighbor will be increased. After processing all the training instances, the algorithm discards those examples with accuracy lower than a certain threshold. As center instances are usually neighbors of other instances from the same class, they generally gain high accuracy, thus being retained by ACC.

A slight modification of the original Wilson's algorithm consists of using, instead of the k-NN classifier, an alternative rule based on the k nearest centroid neighbors (k-NCN) [26], which has been demonstrated to be superior to the traditional k-NN classifier in many practical situations. This kind of neighborhood is defined taking into account not only the proximity of instances to a given input pattern, but also their symmetrical distribution around it.

Let p be a given point whose k nearest centroid neighbors should be found in a given TS. These k neighbors can be searched for through an iterative procedure in the following way [8]:

1. The first nearest centroid neighbor of p corresponds to its nearest neighbor, say q_1.

2. The i-th nearest centroid neighbor, say q_i ($i \geq 2$), is such that the centroid of this and all previously selected nearest centroid neighbors, $q_1, \ldots, q_i$ is the closest to p.

2.2 Editing by Estimating Class Conditional Probabilities

Recently, new editing schemes have been proposed, in which the elimination rule is based on an estimation of the probability of each training instance to belong to a certain class, that is, considering the form of the underlying probability distribution in the neighborhood of a point [29]. In order to estimate the values of these distributions, we can compute the distance between a given sample and the training instances.

Given a sample, the closer an instance, the more likely this sample belongs to the same class as the one of such an instance. Accordingly, let us define the probability $P_i(x)$ that a sample x belongs to a class i as:

$$P_i(x) = \sum_{j=1}^{k} p_i^j \frac{1}{1 + \delta(x, x^j)} \qquad (1)$$

where p_i^j denotes the probability that the j-th nearest neighbor x^j belongs to class i, and δ represents a certain distance function. Initially, the values of p_i^j for each instance are set to 1 for its class label assigned in the TS, and 0 for the other classes.

The meaning of the above expression states that the probability that a sample x belongs to a class i is the weighted average of the probabilities that its k nearest neighbors belong to that class. The weight is inversely proportional to the distance from the sample to the corresponding k nearest neighbors. After normalizing, we obtain:

$$P_i^N(x) = \frac{P_i(x)}{\sum_{j=1}^{L} P_j(x)} \qquad (2)$$

where L represents the number of classes.

From this, we can derive a new decision rule, namely k-Prob, in which a new sample x will be assigned to the class whose probability $P_i^N(x)$ is maximum.

Following the general scheme of Wilson's editing, the new algorithms consist of eliminating from the TS those instances whose label does not coincide with that assigned by the decision rule based on class conditional probabilities (k-Prob).

A further extension to this proposal consists of considering a threshold, $0 < \mu < 1$, in the classification rule, with the aim of eliminating those instances whose probability to belong to the class assigned by the rule is not significant. Correspondingly, we are removing samples from the TS that are in the decision borders, where the class conditional probabilities overlap and are confusing, in order to obtain edited sets whose instances have a high probability of belonging to the class assigned in the TS.

3 Adding Unlabelled Data on the Training Set

A basic goal of the learning system presented in this paper is to make it as automatic as possible. Accordingly, the procedure has been designed to work by incorporating new patterns into the TS after they have been labelled by the own system (without the participation of a human expert). However, it is evident that this working method can be self-defeating, in the sense that these new training elements will have the class label directly assigned by the decision rule. Therefore, there is the risk to add several mislabelled cases on the TS and consequently, to degrade the overall system accuracy. The system we have designed attempts to overcome such a difficulty by employing some editing algorithms.

On the other hand, albeit the original training instances are generally labelled by human experts (or, at least, under their supervision), it is still possible to introduce errors into the initial TS. Correspondingly, our first task will consist of looking for outliers (noisy, atypical and mislabelled patterns) in the TS in order to obtain a collection of correctly labelled instances.

In summary, the learning procedure for partially supervised domains will consist of the following steps:

1. Initial TS is stored in memory.
2. A first filter is applied to the original TS in order to remove possible noisy instances. As a by-product, it also produces a certain reduction in the TS size. The resulting edited set will be here referred to as *base knowledge*.
3. Classification phase starts with the base knowledge working as the current TS.
4. The set of new labelled patterns (those classified during the previous step) is now edited in order to detect possible misclassifications. The patterns identified as erroneous by the filtering algorithm will be removed from that set.
5. The base knowledge is now updated by incorporating the new labelled patterns that have not been discarded in the previous step.
6. Return to Step 3 with the new base knowledge.

For the filters considered in this procedure, one could employ any editing algorithm present in the literature. For example, in this paper, we have applied two of the schemes introduced in Section 2: the k-NCN editing, and the first algorithm based on class conditional probabilities, namely WilsonProb [29]. Analogously, the classification phase (Step 3) can be performed by applying any classifier. In particular, here we have tested the classical k-NN decision rule, the k-NCN classifier, and the new k-Prob classification scheme.

Note that the original base knowledge (i.e., the initial TS) constitutes the only supervised element of our learning system. The unsupervised component comes from the unlabelled patterns that are sequentially classified (Step 3) and edited (Step 4) by the own system.

3.1 Related Works

The EM-based algorithms are among the most widely-used in semi-supervised learning. EM iterates between estimating model parameters and inferring soft labels for unlabelled points. It assumes a generative model (typically a mixture of Gaussians) and has been applied to text classification [23] and face pose determination [1]. Unfortunately, when the data does not match EM's generative assumptions, the algorithm goes astray, and the information from labelled data is overwhelmed by unlabelled patterns

Dasarathy [12] proposed a decision system with a design very related to ours. He was also concerned with the robustness of the system through varying domains and with the problem of unrepresentative pre-training. The latter is what he called "partially exposed environments". Consequently, Dasarathy presented an on-line adaptive learning system with two capabilities: a) to progressively improve the classification of patterns belonging to the known classes and, b) to detect the objects not belonging to the currently known classes

However, Dasarathy's system requires the steady participation of a human expert to be in charge of the evaluation of the labels assigned by the system to new patterns and to decide which of them are to be incorporated into the TS. Unfortunately,

in real-world operational phase, such operator supervision may be unavailable. We avoid this bottleneck by including in our procedure the necessary tools to allow the system to decide which pieces of new knowledge are trustworthy enough to be accepted.

More recently, Bennett et al. [4] introduced an adaptive semi-supervised ensemble method that constructs classification ensembles based on both labelled and unlabelled data. The algorithm alternates between assigning "pseudo-classes" to the unlabelled data using the existing ensemble and constructing the next base classifier using both the labelled and pseudolabeled data.

Similarly, Zhou and Li [31] proposed a semi-supervised learning algorithm, namely tri-training, which generates three classifiers from the original labelled example set. These classifiers are then refined using unlabelled examples in the tri-training process. In detail, in each round of tri-training, an unlabelled example is labelled for a classifier if the other two classifiers agree on the labelling, under certain conditions.

4 Experimental Set-Up

In the experimental study, we have included six data sets taken from the UCI Machine Learning Database Repository (`http://www.ics.uci.edu/~mlearn`). To increase statistical significance of the results in domains with a limited number of examples, the 5-fold cross validation technique has been applied to the experiments in this paper. About 80% out of the total number of patterns available has been used for training and the rest for a test set. The results reported here correspond to the average over the five random partitions.

For each fold, the training patterns were randomly divided into a number of partitions, all keeping the a priori class probabilities. One of these random partitions is used as the initial TS, while the rest will be employed as sets of unlabelled data in order to simulate the sequence required for developing the capability of increasing the knowledge by means of the algorithm introduced in the previous section.

The main characteristics of the data sets used in the present experiments are summarized in Table 1. The seventh column (Partitions) indicates the number of random partitions produced for each database. This means that for example, in Breast database the classification system will have eight opportunities to increase its base knowledge, that is, the number of sets with unlabelled data (besides the initial TS). By this, it is evident that the amount of labelled instances is much smaller than that of the unlabelled patterns. The reason is that, as already stated in Section 1, in real applications collecting labelled examples often becomes a costly and difficult process, thus we are here reproducing this practical situation.

The experiments consist of comparing the 1-NN classification accuracy when using the initial TS with that obtained when incorporating the new labelled patterns to the TS after processing each of the partitions. The aim is to evaluate the capability of increasing the knowledge with the application of our learning algorithm in a partially supervised environment. As a baseline, the last column in Table 1 provides the 1-NN classification accuracy when using the original TS (note that this corresponds to one individual partition) without any editing.

5 Empirical Results

Table 2 and Table 3 provide the classification accuracies over the different databases used in the present experiments. The first column (t) refers to each partition included in the process. Thus $t = 0$ represents the initial base knowledge, that is, the original TS after being edited. The set obtained at any time $t > 0$ is then incorporated into the previous knowledge (the set of instances available at time $t-1$). For example, in $t = 1$ the current knowledge refers to that acquired in $t = 0$, and it is now employed to classify the first set of unlabelled patterns. After classifying, we edit the new labelled instances in order to discard possible misclassifications. Then, the current knowledge is updated by including the instances that have not been eliminated during the editing stage. The result will correspond to the input set in $t = 2$.

The meaning of Alg1, Alg2, Alg3, and Alg4 in Tables 2– 4 is as follows. In the case of Alg1, we have employed the k-NCN algorithm for editing and the k-NN rule for classification. Alg2 uses the k-NCN algorithm both for editing and for classifying new patterns. Alg3 applies Wilson-Prob for editing and the k-Prob decision rule for classifi-

cation. Finally, Alg4 is equal to Alg3, but using the nearest centroid neighborhood instead of the classical nearest neighborhood. Values in bold type indicate the first occurrence of the highest accuracy for each algorithm and each database.

From the results reported in Tables 2– 4, some conclusions can be drawn. First, it has to be noted that all implementations outperform the 1-NN classification accuracy reported as a baseline. On the other hand, except Alg3 when applied over Diabetes, German and Satimage databases, all the other cases show a certain improvement in performance with respect to the original edited TS ($t = 0$). From this observation, Alg3 appears as the alternative with the poorest results. Nonetheless, in terms of accuracy, it is not straightforward to decide which learning algorithm is the best. In practice, any of those three algorithms (Alg1, Alg2, and Alg4) could constitute a good solution for increasing the knowledge in a partially supervised environment.

It is worth pointing out the fact that in general, the system obtains a maximum value in performance after processing a relatively small number of partitions. This is important because it can mean that after a number of iterations, the inclusion of more instances does not provide more information to the system. In this situation, the system increases the size of the TS, but without increasing the useful knowledge. This is a crucial issue that will be investigated in further extensions to this work.

6 Conclusions and Further Extensions

In this paper, a learning algorithm to increase the knowledge in partially supervised environments has been introduced. It makes use of a reduced number of labelled instances and a (possibly) large amount of unlabelled patterns. The system includes a set of tools allowing to filter the new knowledge acquired during operation. Thus we pursue to avoid the risk of incorporating several mislabelled patterns into the TS and consequently, to degrade the overall system performance.

In the empirical evaluation of the learning system, we have used different classification rules and several editing algorithms. Except in the case of employing a scheme based on class conditional probabilities for both classification and editing (Alg3), all the other alternatives have been demonstrated to perform well enough for increasing the knowledge (reducing the classification error).

An important issue related to the performance of a system with the capability of increasing its knowledge refers to the possibility for the TS size to grow too much and consequently, some problems related to storage space and classification time can make such a system useless. Although editing has the property, as a by-product, of reducing the TS size, this is not achieved in a considerable amount. Accordingly, possible extensions to this work are in the direction of including some techniques to intelligently reduce the TS size. To this end, both adaptive and selective condensing algorithms [18] can be of interest to control the TS size.

Also, the possibility of discovering new classes not present in the original TS can result especially important for this kind of learning systems in partially supervised domains. Therefore, future research includes the study of unsupervised (clustering) techniques in order to incorporate this additional capability into the learning system here presented.

Acknowledgments

This work has been partially supported by grant TIC2003-08496 from the Spanish CICYT (Ministerio de Ciencia y Tecnología). We would like to thank the anonymous Reviewers and the Guest Editors of this Special Issue for their valuable and constructive remarks.

References

[1] S. Baluja. Using labeled and unlabeled data for probabilistic modeling of face orientation, *International Journal of Pattern Recognition and Artificial Intelligence*, 14:1097–1107, 2000.

[2] R. Barandela and E. Gasca. Decontamination of training data for supervised pattern recognition methods, *In Advances in Pattern Recognition, Lecture Notes in Computer Science*, pages 612–620, 2000.

[3] M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds, *Machine Learning*, 56:209–239, 2004.

[4] K.P. Bennett, A. Demiriz, and R. Maclin. Exploiting unlabeled data in ensemble methods, *In Proc. 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pages 289–296, 2002.

[5] A.M. Bensaid, L.O. Hall, J.C. Bezdek, and L.P. Clarke. Partially supervised clustering for image segmentation, *Pattern Recognition*, 29:859–871, 1996.

[6] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts, *In Proc. 18th. Intl. Conf. on Machine Learning*, pages 19–26, 2001.

[7] V. Castelli and T.M. Cover. On the exponential value of labeled samples, *Pattern Recognition Letters*, 16:105–111, 1995.

[8] B.B. Chaudhuri. A new definition of neighborhood of a point in multi-dimensional space, *Pattern Recognition Letters*, 17:11–17, 1996.

[9] T.M. Cover. Estimation by the nearest neighbor rule, *IEEE Trans. on Information Theory*, 14:50–55, 1968.

[10] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification, *IEEE Trans. on Information Theory*, 13:21–27, 1967.

[11] B.V. Dasarathy. *Nearest Neighbor Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, Los Alamos, CA (1991).

[12] B.V. Dasarathy. Adaptive decision systems with extended learning for deployment in partially exposed environments, *Optical Engineering*, 34:1269–1280, 1995.

[13] P.A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice Hall, Englewood Cliffs, NJ (1982).

[14] F.J. Ferri, J.V. Albert, and E. Vidal. Considerations about sample-size sensitivity of a family of edited nearest-neighbor rules, *IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics*, 29:667–672, 1999.

[15] K. Hattori and M. Takahashi. A new edited k-nearest neighbor rule in the pattern classification problem, *Pattern Recognition*, 33:521–528, 2000.

[16] N. Japkowicz and S. Stephen. The class imbalance problem: a systematic study, *Intelligent Data Analysis*, 6:429–449, 2002.

[17] C.-K. Keung and W. Lam. Prototype generation based on instance filtering and averaging, *In Proc. 4th. Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, pages 142–152, 2000.

[18] S.-W. Kim and B.J. Oommen. A brief taxonomy and ranking of creative prototype reduction schemes, *Pattern Analysis & Applications*, 6:232–244, 2003.

[19] J. Koplowitz and T.A. Brown. On the relation of performance to editing in nearest neighbor rules, *Pattern Recognition*, 13:251–255, 1981.

[20] M.A. Krogel and T. Scheffer. Multirelational learning, text mining, and semi-supervised learning for functional genomics. *Machine Learning*, 57:61–81, 2004.

[21] L.I. Kuncheva. Editing for the k-nearest neighbors rule by a genetic algorithm, *Pattern Recognition Letters*, 16:809–814, 1995.

[22] P. Mantero, G. Moser, and S.B. Serpico. Partially supervised classification of remote sensing images through SVM-based probability density estimation, *IEEE Trans. on Geoscience and Remote Sensing*, 43:559–570, 2005.

[23] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM, *Machine Learning*, 39:103–134, 2000.

[24] S. Okamoto and N. Yugami. Effects of domain characteristics on instance-based learning algorithms, *Theoretical Computer Science*, 298:207–233, 2003 .

[25] J.S. Sánchez, F. Pla, and F.J. Ferri. Prototype selection for the nearest neighbour rule through proximity graphs, *Pattern Recognition Letters*, 18:507–513, 1997.

[26] 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*, 24:1015–1022, 2003.

[27] M.O. Szummer. *Learning from Partially Labeled Data*. PhD thesis, Massachusetts Inst. of Technology, 2002.

[28] I. Tomek. An experiment with the edited nearest neighbor rule. *IEEE Trans. on Systems, Man and Cybernetics*, 6:448–452, 1976.

[29] F. Vazquez, J.S. Sánchez, and F. Pla. A stochastic approach to Wilson's editing algorithm, *In Pattern Recognition and Image Analysis, Lecture Notes in Computer Science*, pages 35–42, 2005.

[30] D.L. Wilson. Asymptotic properties of nearest neighbor rules using edited data sets, *IEEE Trans. on Systems, Man and Cybernetics*, 2:408–421, 1972.

[31] Z.-H. Zhou and M. Li. Tri-Training: Exploiting unlabeled data using three classifiers, *IEEE Trans. on Knowledge and Data Engineering*, 17:1529–1541, 2005.

	Classes	Features	Size	% Class 1	% Class 2	Partitions	NN Accuracy
Breast	2	9	683	65.2	34.8	9	92.48
Diabetes	2	8	786	34.9	65.1	10	66.32
German	2	24	1002	70.4	29.6	13	65.81
Heart	2	13	270	55.6	44.4	8	53.33
Satimage	6	36	6453			18	81.01
Texture	11	40	5500			16	91.74

Table 1: A brief summary of the UCI databases used in the experiments. The a priori probabilities in the Satimage database are 23.8%, 10.9%, 21.1%, 9.7%, 11.0%, and 23.5%. In the case of Texture, each class contains 500 patterns.

	Breast				Diabetes			
t	Alg1	Alg2	Alg3	Alg4	Alg1	Alg2	Alg3	Alg4
0	92.54	92.54	92.54	92.54	66.67	66.67	**70.24**	68.45
1	94.03	94.03	**94.03**	94.03	66.07	66.07	70.24	69.05
2	94.03	94.03	94.03	94.03	66.07	68.45	69.64	69.64
3	94.03	94.03	94.03	94.03	66.07	69.64	67.86	69.64
4	**95.52**	94.03	92.54	**95.52**	65.48	69.05	67.26	69.64
5	95.52	94.03	92.54	95.52	65.48	69.05	67.86	69.64
6	95.52	94.03	92.54	95.52	66.07	69.64	67.86	69.64
7	95.52	**95.52**	94.03	95.52	66.67	70.24	67.86	70.24
8	95.52	95.52	94.03	95.52	**67.26**	**70.83**	67.86	**70.83**
9					67.26	68.45	66.67	70.24
1-NN	92.48				66.32			

Table 2: Classification accuracies for Breast and Diabetes databases (1-NN indicates the classification accuracy when using the original TS without any editing).

	German				Heart			
t	Alg1	Alg2	Alg3	Alg4	Alg1	Alg2	Alg3	Alg4
0	67.61	67.61	**71.83**	69.01	51.61	51.61	54.84	54.84
1	69.01	69.01	71.83	69.01	61.29	58.06	58.06	61.29
2	**70.42**	**70.42**	71.83	69.01	61.29	**64.52**	**64.52**	61.29
3	70.42	70.42	71.83	69.01	64.52	64.52	64.52	64.52
4	70.42	70.42	70.42	69.01	**67.74**	64.52	64.52	64.52
5	70.42	70.42	67.61	**70.42**	67.74	64.52	64.52	64.52
6	67.61	70.42	67.61	70.42	67.74	64.52	64.52	64.52
7	67.61	70.42	69.01	70.42	67.74	64.52	64.52	**67.74**
8	67.61	70.42	69.01	70.42				
9	67.61	70.42	69.01	70.42				
10	67.61	70.42	70.42	70.42				
11	67.61	70.42	70.42	70.42				
12	67.61	70.42	70.42	70.42				
1-NN	65.81				53.33			

Table 3: Classification accuracies for German and Heart databases (1-NN refers to the classification accuracy when using the original TS without any editing).

	Satimage				Texture			
t	Alg1	Alg2	Alg3	Alg4	Alg1	Alg2	Alg3	Alg4
0	83.01	85.62	**85.95**	84.64	93.12	93.16	93.44	93.44
1	83.01	86.28	85.95	84.64	95.41	95.74	94.75	94.75
2	83.33	**85.95**	85.29	**85.29**	**95.41**	95.74	**95.08**	94.43
3	83.66	85.62	84.97	83.99	93.77	95.41	93.44	92.79
4	83.66	85.29	84.64	83.99	93.77	95.41	93.77	93.77
5	84.31	84.64	84.31	83.01	94.75	**96.39**	94.43	94.43
6	84.64	84.31	84.31	81.70	94.75	95.74	94.43	94.43
7	84.31	83.33	83.01	81.05	94.75	95.74	94.43	**95.08**
8	84.31	83.01	83.66	80.72	94.10	95.08	94.10	95.08
9	84.31	82.68	83.66	80.07	94.10	95.08	93.77	94.75
10	84.97	83.33	83.66	81.05	94.43	95.08	93.44	94.75
11	84.97	83.66	83.99	81.05	94.10	95.08	93.44	94.75
12	**85.62**	83.66	84.97	80.72	93.77	95.08	93.77	94.75
13	85.29	83.01	84.97	80.39	93.77	95.41	94.10	94.75
14	85.29	82.68	85.29	81.05	93.77	95.41	94.10	94.75
15	85.29	82.35	85.29	81.37	94.43	95.41	94.75	94.43
16	85.29	82.35	84.64	81.05				
17	85.29	82.35	84.64	80.72				
1-NN	81.01				91.74			

Table 4: Classification accuracies for Satimage and Texture databases (1-NN is the classification accuracy when using the original TS without any editing).