
PERGAMON

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition 37 (2004) 1561–1564

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Rapid and Brief Communication

High training set size reduction by space partitioning and prototype abstraction

J.S. Sánchez*

Department of Llenguatges i Sistemes Informàtics, Universitat Jaume I, Campus Riu Sec, Castelló E-12071, Spain

Received 17 December 2003; accepted 23 December 2003

Abstract

Instance-based learning methods like the nearest neighbour classifier generally suffer from the indiscriminate storage of all training instances, resulting in large memory requirements and slow execution speed. In this paper, new training set size reduction methods based on prototype generation and space partitioning are proposed. Experimental results show that the new algorithms achieve a very high reduction rate with still an important classification accuracy.

Keywords: Nearest neighbour; Set size reduction; Prototype generation; Space partitioning

1. Introduction

Many supervised learning algorithms use a collection of training instances, typically called a training set (TS), to estimate the class label of new input vectors. Each example in the TS has an attribute vector and a class label (i.e., the output value). After learning from the TS, the algorithm is presented with additional input vectors and must use some inductive bias to decide the output value. Methods that employ this technique are also known as instance-based learners, lazy learners, case-based learners, and exemplar-based learners [1].

The nearest neighbour (NN) algorithm [2] is one of the most widely studied examples of instance-based methods. During classification, the NN algorithm employs an appropriate distance metric defined on the feature space to determine how close a new input vector x is to each instance in the TS, and then uses the nearest case to predict the class label of x. An improved version of this algorithm corresponds to the k-NN rule, which consists of assigning a new input vector to the class most frequently represented among the k closest training instances.

In general, these learners must decide which instances to store in the TS in order to avoid excessive storage and time complexity, and possibly to improve classification accuracy by avoiding noise and overfitting [3]. For example, the instances employed to train the NN classifier are stored indiscriminately. This means that the NN rule has relatively large memory requirements: it needs to search through all available instances to classify a new input vector, so it can become too slow during classification. On the other hand, since it stores every instance in the TS, noisy cases are also stored, possibly degrading significantly the classification accuracy.

Among the many proposals to reduce the storage requirements and time complexity of the NN algorithm (three extensive surveys can be found in [2–4]), it is worth mentioning those that try to obtain a more efficient classification scheme by defining a reduced TS. In this context, a possible taxonomy of TS size reduction techniques considers two main groups by distinguishing between *selection* and *abstraction* methods. While the former family consists of picking a subset of the original training instances, the abstraction (or generation) group builds new artificial prototypes summarizing a number of similar instances.

One problem with using the original instances is that they assume that optimal examples can be found in the original

* Tel.: 34-964-728350; fax: 34-964-728435.
E-mail address: sanchez@uji.es (J.S. Sánchez).

0031-3203/$30.00
doi:10.1016/j.patcog.2003.12.012

set but, in fact there may not be any vector located at the precise points that would make the most accurate learning algorithm. Thus, prototypes can be artificially generated to exist exactly where they are needed, if such locations can be accurately determined. This paper focuses on the problem of reducing the TS size while trying to preserve classification accuracy by means of new abstraction (or generation) methods, which are based on three different heuristics for feature space partitioning.

2. New TS size reduction algorithms

The generation algorithms proposed here are based on different heuristics for partitioning the feature space, along with several techniques for defining prototypes. They are initially inspired by the technique proposed by Chen and Józwik [5], but trying to avoid drastic changes in the form of the decision boundaries associated with the original TS, which constitutes one of the main drawbacks related to that algorithm.

These reduction schemes will be called RSP1–RSP3 (RSP, reduction by space partitioning). From now on, the original TS will be denoted by T, while S will refer to the resulting reduced set. All these algorithms consist of dividing the TS into a number of subsets by taking into account its diameter.

Definition 1. The diameter of a set X, say Θ_X, is defined as the distance between its two farthest points.

As in the case of most generation methods, the RSP approaches require to preprocess the original TS using an editing algorithm, in order to eliminate atypical and mislabelled instances as well as to smooth the decision boundaries.

2.1. RSP1: diameter of a set

One problem associated with the heuristic introduced by Chen and Józwik refers to the fact that in some practical situations, all instances that belong to one of the classes can be discarded from the TS because of the way in which the representatives are determined. Consequently, if one class is emptied, the learner will fail on all cases that belong to such an eliminated class.

Taking care of this problem, the first algorithm proposed here starts by computing the diameter of T (given by its two farthest points, say p_1 and p_2) and dividing the TS into two blocks. One subset contains those instances that are nearer p_1 than p_2, while the other block contains the remaining cases. This process is repeated until obtaining b subsets. Then, for each block, RSP1 computes one centroid for each different class existing in the subset. As a result, we will obtain c prototypes per subset, being c the number of different classes covered by such a partition. Algorithmically, RSP1 can be written as follows:

1. Let $b_c = 1$ (b_c is the current number of subsets in T), and $i = 1$.
2. Let $B = T$.
3. Find the two farthest points, p_1 and p_2, in B.
4. Divide the set B into two subsets B_1 and B_2, where
$$B_1 = \{p \in B : d(p, p_1) \leqslant d(p, p_2)\},$$
$$B_2 = \{p \in B : d(p, p_2) < d(p, p_1)\}.$$
5. Let $b_c = b_c + 1$, $C(i) = B_1$, and $C(b_c) = B_2$.
6. Let $I_1 = \{i : C(i)$ contains instances from two different classes at least$\}$, and let $I_2 = \{i : i \leqslant b_c\} - I_1$.
7. Let $I = I_1$ if $I_1 \neq \emptyset$ else $I = I_2$.
8. Find the two farthest points, $q_1(i)$ and $q_2(i)$, in each $C(i)$ for $i \in I$.
9. Find the set $C(j)$ with the largest diameter Θ_j.
10. Let $B = C(j)$, $p_1 = q_1(j)$, and $p_2 = q_2(j)$.
11. If $b_c < b$ (b is the number of final subsets) then go to 4.
12. Find the centroids $c(l, i)$ for each class l in subset $C(i)$, $i = 1, 2, \ldots, b$.
13. Put all $c(l, i)$ in the resulting set S.

For this first reduction technique, the number of final representatives in S cannot be determined in advance. On the contrary, the number of blocks is given, which results in a number of prototypes $b \leqslant |S| \leqslant bm$, where b denotes the number of subsets to create and m is the number of classes present in the original TS.

It is important to note that by defining the centroid of instances from each class, it is expected that the decision boundaries associated with the resulting set of prototypes will not be displaced as much as in the case of Chen's algorithm. On the other hand, even more important, this heuristic guarantees that no class is empty after applying the algorithm.

2.2. RSP2: overlapping degree

In the previous algorithm, the partition criterion can be stated as follows: divide the subset with the largest diameter (Steps 9 and 10 of RSP1). The idea is that the largest subset should probably contain more instances than any other block and therefore, we should also obtain the highest reduction rate. However, theory dictates that instances from a class should be as close to each other as possible, while instances belonging to different classes should be located as far as possible. Accordingly, from a theoretical point of view, it could be more appropriate to split the subset with the highest overlapping degree among instances from distinct classes.

Definition 2. The overlapping degree of a set X, say Φ_X, is defined as the ratio of the average distance between instances belonging to different classes, $D^{\neq}$, and the average distance between instances that are from the same class, $D^{=}$.

As in the case of RSP1, one prototype for each different class present in the resulting subsets is computed. The input to the algorithm is also the number of partitions to obtain and therefore, the number of resulting instances will be the same as that of RSP1.

2.3. RSP3: class homogeneity

The third reduction heuristic consists of performing partitions until all subsets are class homogeneous. The rationale behind this is that each block should represent a cluster of instances belonging to only one class. In this case, it is not necessary to provide any tuning parameter (number of blocks or number of final prototypes) to the algorithm.

Definition 3. A set X is said to be class homogeneous if it does not contain a mixture of instances that belong to different classes.

RSP3 can employ both split criteria defined previously: divide the subset with the largest diameter or that with the highest overlapping degree. In fact, the partition criterion is not important here because all heterogeneous subsets have to be finally divided. The number of prototypes generated by this approach will be generally much higher than that of any other RSP scheme, especially if mislabelled and atypical instances have not been previously removed from the TS.

3. Experimental results and discussion

The reduction techniques just introduced have been empirically compared with Wilson's, Hart's, and Chen's algorithms. The basic 1-NN classification rule with 100% of training instances has also been included as base line. To keep the research conveniently focused, we have worked with nine data sets, taken from the UCI Database Repository, where all attributes are continuous. For each data set, five-fold cross-validation has been used to estimate the average classification accuracy and reduction rate. Experiments consist of applying the 1-NN rule to each of the test sets, where the training portion has been preprocessed by means of some reduction algorithm.

For each database, we have firstly applied Wilson's editing to the original TS in order to remove mislabelled instances and smooth the decision boundaries. Afterwards, Hart's and Chen's algorithms along with RSP methods have been used on the Wilson's edited set to reduce the number of training examples. In the case of those methods that generate new prototypes, in order to compare all schemes when containing approximately the same number of prototypes, firstly, RSP1 and RSP2 have been tested for several values of the parameter b (number of subsets) and then, Chen's algorithm has been applied according to the average number of prototypes provided by RSP1 and RSP2.

From the results reported in Table 1, some preliminary comments can be drawn. As expected, 1-NN and Wilson's algorithms present the highest classification accuracy almost

Table 1
Classification accuracy (Acc.) and reduction rate (Size) for each data set

		1-NN	Wilson's	Hart's	Chen's	RSP1	RSP2	RSP3
Cancer	Acc.	94.53	94.89	91.24	95.01	94.53	94.28	94.28
	Size	—	3.02	96.70	99.01	98.72	98.63	96.89
Clouds	Acc.	84.60	88.42	88.08	58.22	65.61	68.35	85.52
	Size	—	12.20	94.76	99.79	99.73	99.68	74.58
Glass	Acc.	72.50	66.25	67.50	37.50	50.83	54.17	63.75
	Size	—	35.63	87.36	96.93	95.89	95.08	69.83
Heart	Acc.	59.26	64.81	66.67	63.27	66.67	64.81	62.04
	Size	—	36.11	86.11	97.53	96.72	96.29	74.07
Liver	Acc.	68.12	70.29	63.77	57.00	61.11	60.63	67.39
	Size	—	36.23	80.80	97.74	96.68	96.56	64.86
Phoneme	Acc.	76.08	72.95	70.14	65.81	65.68	68.07	72.19
	Size	—	43.55	79.12	99.68	99.66	99.69	82.93
Pima	Acc.	63.40	68.96	67.05	65.47	67.87	70.59	69.94
	Size	—	29.84	84.36	99.13	98.43	98.59	81.14
Satimage	Acc.	79.98	79.67	78.24	64.15	75.41	76.01	78.82
	Size	—	6.06	77.38	99.53	99.55	99.59	75.02
Wine	Acc.	73.53	73.54	71.39	65.69	67.16	66.18	73.53
	Size	—	30.21	83.91	96.20	95.75	95.63	87.15
Average	Acc.	74.67	75.53	73.79	63.57	68.32	69.23	74.16
	Size	—	25.87	85.61	98.39	97.90	97.75	78.49

without exception for all the data sets, but it is mainly due to retaining all or most of the prototypes (Wilson's algorithm only removes 25.87% of the original training cases). However, in general, all RSP methods achieve higher classification accuracy than Chen's technique, with practically the same reduction rate.

Hart's results are quite similar to those of RSP3: a sufficiently high reduction percentage without an important degradation in accuracy. In fact, for many practical situations, the RSP3 algorithm along with Hart's condensing could be the best reduction schemes in terms of balancing accuracy with storage reduction. However, if a particular application needs to obtain the highest reduction rate, then RSP1, RSP2, and Chen's schemes provide an appropriate solution because they show an extremely high reduction percentage (about 98.00%), although they yield moderate classification performance.

When comparing the three approaches proposed here, RSP3 clearly shows the highest classification accuracy and it still removes many training instances (78.49% in average), although it runs substantially slower than any other RSP algorithm (it has to iterate until all subsets become class homogeneous). On the other hand, RSP1 and RSP2 eliminate close to 98% of the prototypes and the average accuracy is over 68%.

Finally, it is to be noted that RSP3 achieves a classification rate close enough to that of Wilson's (differences between them are not statistical significant in seven out of the nine databases), but with a very important difference in the reduction percentage (52.62% higher in average).

Acknowledgements

This work has partially been supported by Grants TIC2003-08496-C04-03 from Spanish CICYT and P1-1B2002-07 from Fundació Caixa Castelló-Bancaixa.

References

[1] D.W. Aha (Ed.), Lazy Learning, Kluwer Academic Publishers, Norwell, MA, 1997.

[2] B.V. Dasarathy, Nearest Neighbor Norms: NN Pattern Classification Techniques, IEEE Computer Society Press, Los Alamitos, CA, 1991.

[3] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Mach. Learning 38 (2000) 257–286.

[4] J.C. Bezdek, L.I. Kuncheva, Nearest prototype classifier designs: an experimental study, Int. J. Intell. Systems 16 (2001) 1445–1473.

[5] C.H. Chen, A. Józwik, A sample set condensation algorithm for the class sensitive artificial neural network, Pattern Recognition Lett. 17 (1996) 819–823.