Space Partitioning for Instance Reduction in Lazy Learning Algorithms *

M.C. Ainslie and J.S. Sánchez

Dept. de Llenguatges i Sistemes Informàtics, Universitat Jaume I,
Campus Riu Sec, E-12071 Castelló, Spain

Abstract. Lazy learning methods suffer from the indiscriminate storage of all training instances, resulting in large memory requirements and slow execution speed. This paper focuses on the problem of reducing the training set size by presenting new instance selection schemes.

1 Introduction

Many supervised learning algorithms use a collection of training instances, typically called training set (TS), to estimate the class label of new input vectors. Each instance in the TS has an attribute vector and a class label (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 lazy learners.

The nearest neighbour (NN) algorithm [5] is one of the most widely studied examples of lazy learning methods. During classification, the NN scheme 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 uses the nearest instance to predict the class of x. An improved version of this corresponds to the k-NN rule, which consists in assigning a new input vector to the class most frequently represented among the k closest instances stored in the TS.

In general, lazy learning algorithms must decide which instances to store in the TS for use during classification in order to avoid excessive storage and time complexity, and possibly to improve classification accuracy by avoiding noise and overfitting. For example, the instances used to train the NN classifier are stored indiscriminately. This means that the NN approach has to search through all available cases 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 instances are also stored, possibly degrading significantly the accuracy.

Among the many proposals to reduce the storage requirements and time complexity of the NN rule, it is worth mentioning those that try to obtain a more efficient scheme by removing some instances from the TS. In this context, it is possible to differentiate between two main types of TS reduction techniques: those that retain a subset of the original instances [1,6,8] and, those that modify the training instances using a new representation [3,4].

* Partially supported by grant No. TIC2000-1703-C03-03 from the Spanish CICYT.

2 Training Set Size Reduction Techniques

In lazy learning, the problem of instance selection is primarily related to instance deletion as irrelevant and harmful cases are removed while retaining only critical instances. Others modify the instances themselves to reduce the TS size.

2.1 Reduction by Eliminating Instances

Hart's algorithm [6] is the earliest attempt at minimizing the number of stored instances by retaining only a *consistent* subset of the original TS. A consistent subset, S, of a TS, T, is some subset that correctly classifies every instance in T using the 1-NN rule. One generally is interested in the *minimal* consistent subset to minimize the cost of storage and computing time. Hart's condensing does not guarantee finding the minimal subset and furthermore, different subsets are given when the TS order is changed.

Aha et al. [1, 2] presented the incremental learning schemes IB1-IB4. With IB2, if a new case to be added can already be correctly classified by the current training instances, then the case is discarded and not stored at all. IB3 addresses the problem of keeping noisy instances by retaining only acceptable misclassified cases. IB4 is thought to handle irrelevant attributes by building a set of attribute weights for each class

Wilson [8] introduced the first editing proposal. Briefly, this consists in using the k-NN rule to estimate the class of each instance in the TS, and removing those whose class label does not agree with that of the majority of its k neighbours. This algorithm tries to eliminate mislabelled instances from the TS as well as close border instances, smoothing the decision boundaries.

2.2 Reduction by Generating Prototypes

Some algorithms try to modify the instances in order to reduce the TS size, instead of deciding which ones to retain. Some of them artificially generate prototypes in locations accurately determined.

Chang [3] introduced an algorithm in which each instance in the TS is initially considered as a prototype. Then, it consists in repeatedly attempting to merge the nearest two existing instances into a new single prototype. Two instances p and q are merged only if they are from the same class and, after replacing them with prototype z, the consistency property can be guaranteed.

Chen and Józwik [4] proposed an algorithm which consists in dividing the TS into subsets using the concept of *diameter of a set* (i.e., the distance between its two farthest points). The algorithm starts by partitioning the TS into two subsets by the middle point between the two farthest cases. The next division is performed for the subset that contains a mixture of instances from different classes. If more than one subset satisfies this condition, then that with the largest diameter is divided. The number of partitions will be equal to the number of instances initially defined. Finally, the subsets are replaced by their centroids, which will assume the same class label as the majority of cases in each subset.

3 Instance Reduction by Space Partitioning

The algorithms developed in this section are directed towards defining prototypes by partitioning the feature space. They will be here called IRSP1-IRSP4 (Instance Reduction by Space Partitioning). From now on, the original TS will be denoted by T, while S will refer to the resulting reduced set.

3.1 IRSP1

One problem associated with the heuristic introduced by Chen and Józwik refers to the fact that, in some cases, all instances from one of the classes can be eliminated from the TS. IRSP1 overcomes this by computing one centroid for each different class existing in the subset. We will obtain c prototypes per subset, being c the number of different classes present in it.

1. IRSP1(T, b):S
2. Let $b_c = 1$ (b_c is the current number of subsets in T), and $i = 1$.
3. Let $B = T$.
4. Find the two farthest points, p_1 and p_2, in B.
5. Divide the set B into two subsets B_1 and B_2, where
   $B_1 = \{p \in B : d(p, p_1) \leq d(p, p_2)\}$
   $B_2 = \{p \in B : d(p, p_2) < d(p, p_1)\}$
6. Let $b_c = b_c + 1$, $C(i) = B_1$, and $C(b_c) = B_2$.
7. Let $I_1 = \{i : C(i)$ contains instances from two different classes at least$\}$, and let $I_2 = \{i : i \leq b_c\} - I_1$.
8. Let $I = I_1$ if $I_1 \neq \emptyset$ else $I = I_2$.
9. Find the two farthest points, $q_1(i)$ and $q_2(i)$, in each $C(i)$ for $i \in I$.
10. Find the set $C(j)$ with the largest diameter.
11. Let $B = C(j)$, $p_1 = q_1(j)$, and $p_2 = q_2(j)$.
12. If $b_c < b$ then go to Step 5.
13. Find the centroids $c(l, i)$ for each class l in subset $C(i)$, $i = 1, 2, \ldots, b$.
14. Put the $c(l, i)$ in the resulting set S.

For this approach, the number of partitions is given, which results in a number of instances greater or equal to the number of subsets and less or equal to the number of partitions multiplied by the number of classes, that is, $b \leq |S| \leq b \cdot m$, where b is the number of subsets and m is the number of classes in the TS.

3.2 IRSP2

Theory dictates that cases from a class would be as close to each other as possible, while instances from different classes would be located as far as possible. Therefore, it seems more appropriate to split the subset by the highest degree of overlapping among instances from different classes. Another difference regarding IRSP1 is that IRSP2 generates only one prototype for each subset and assigns it to the class with a majority of cases.

The overlapping degree of a set A, say O_A, can be defined as the ratio of the average distance between instances belonging to different classes, D_d, and the average distance between instances being from the same class, D_s.

1. IRSP2(T, n):S
2. Let $n_c = 1$ (n_c is the current number of subsets in T), and $i = 1$.
3. Let $B = T$.
4. Find the two farthest points, p_1 and p_2, in B.
5. Divide the set B into two subsets B_1 and B_2, where
   $B_1 = \{p \in B : d(p, p_1) \leq d(p, p_2)\}$
   $B_2 = \{p \in B : d(p, p_2) < d(p, p_1)\}$
6. Let $n_c = n_c + 1$, $C(i) = B_1$, and $C(n_c) = B_2$.
7. Let $I_1 = \{i : C(i)$ contains instances from two different classes at least$\}$, and let $I_2 = \{i : i \leq n_c\} - I_1$.
8. Let $I = I_1$ if $I_1 \neq \emptyset$ else $I = I_2$.
9. Find the set $C(j)$ with the highest overlapping degree, $O_j = \frac{D_d}{D_s}$.
10. Find the two farthest points, $q_1(j)$ and $q_2(j)$, in $C(j)$.
11. Let $B = C(j)$.
12. If $n_c < n$ then go to Step 5.
13. Find the centroids $c(i)$ for each subset $C(i)$, $i = 1, 2, \ldots, n$.
14. Put the $c(i)$ in the resulting set S.

By applying the IRSP2 algorithm, the number of instances in S will be bounded by the number of problem classes and the number of instances given in advance, say n: $m \leq |S| \leq n$.

3.3 IRSP3

IRSP3 corresponds to a combination of IRSP1 and IRSP2. The subset divided at each stage is that with the highest overlapping degree and, one prototype for each different class existing in the resulting subsets is computed. This helps to avoid the possible excessive displacement of the decision boundaries when replacing subsets with a high overlapping degree by only one centroid. The input to the algorithm is the number of partitions to perform and consequently, the number of resulting cases will be the same as that of IRSP1.

3.4 IRSP4

The last reduction heuristic consists in performing partitions until all subsets are homogeneous (no subset contains a mixture of instances from different classes). It is not necessary to provide any tuning parameter (number of partitions or number of instances) to the algorithm. IRSP4 can use both division criteria defined previously. In fact, the partition criterion is not important here because all heterogeneous subsets have to be finally divided.

4 Experimental Results

The schemes introduced in Sect. 3 have been empirically compared with Wilson's, Hart's, and Chen's algorithms. The basic 1-NN rule with 100% of training cases has also been included here for comparison purposes. We have utilized six data sets from the UCI Repository [7], and three from the ELENA Project. We have worked only with domains where all attributes are continuous.

Holdout averaged over five random partitions has been used for each database. A percentage of the instances has been used as the TS and the rest of cases for the test set. Experiments consist in applying the 1-NN rule to the test sets, where the training portion has been preprocessed by some reduction algorithm.

Table 1. Classification accuracy and reduction percentage for each data set.

Data set	1-NN	Wilson's	Hart's	Chen's	IRSP1	IRSP2	IRSP3	IRSP4
Cancer	94.53	94.89	91.24	95.01	94.53	95.13	94.28	94.28
		3.02	96.70	99.01	98.72	99.01	98.63	96.89
Clouds	84.60	88.42	88.08	58.22	65.61	58.87	68.35	85.52
		12.20	94.76	99.79	99.73	99.79	99.68	74.58
Glass	72.50	66.25	67.50	37.50	50.83	41.67	54.17	63.75
		35.63	87.36	96.93	95.89	96.93	95.08	69.83
Heart	59.26	64.81	66.67	63.27	66.67	65.74	64.81	62.04
		36.11	86.11	97.53	96.72	97.53	96.29	74.07
Liver	68.12	70.29	63.77	57.00	61.11	57.00	60.63	67.39
		36.23	80.80	97.74	96.68	97.74	96.56	64.86
Phoneme	76.08	72.95	70.14	65.81	65.68	68.37	68.07	72.19
		43.55	79.12	99.68	99.66	99.68	99.69	82.93
Pima	63.40	68.96	67.05	65.47	67.87	66.23	70.59	69.94
		29.84	84.36	99.13	98.43	99.13	98.59	81.14
Satimage	79.98	79.67	78.24	64.15	75.41	64.00	76.01	78.82
		6.06	77.38	99.53	99.55	99.53	99.59	75.02
Wine	73.53	73.54	71.39	65.69	67.16	65.69	66.18	73.53
		30.21	83.91	96.20	95.75	96.20	95.63	87.15
Average	74.67	75.53	73.79	63.57	68.32	64.74	69.23	74.16
		25.87	85.61	98.39	97.90	98.39	97.75	78.49

Several comments can be made from the results in Table 1. As expected, 1-NN and Wilson's algorithms present the highest average accuracy, but it is mainly due to retaining all or most of the instances (in average, Wilson's algorithm only removes 25.87% of the cases). However, in general, all IRSP methods achieve higher accuracy than Chen's scheme, with practically the same reduction rate. Hart's condensing results are quite similar to those of IRSP4: a sufficiently high reduction degree without an important degradation in accuracy.

Among the algorithms proposed in this paper, IRSP1, IRSP3 and IRSP4 seem to be the best in terms of balancing accuracy with storage reduction. IRSP4

shows the highest classification accuracy and it still removes many instances (78.49% in average). When comparing this approach with Wilson's editing, we can see that IRSP4 achieves an accuracy close enough to that of Wilson's, but with a very important difference in the reduction percentage (52.62% higher in average). On the other hand, IRSP1 and IRSP3 eliminate close to 98% of the cases and the average accuracy is over 68%.

It can be observed that IRSP2 and Chen's algorithms achieve worse accuracy results than the other schemes proposed. It seems that the number of centroids computed for each subset can become more important than the partition criterion used by the algorithm. In fact, in many cases, when applying Chen's method or IRSP2 (i.e., those schemes that compute only one prototype in each subset), all the instances belonging to some classes have been removed from the TS.

5 Concluding Remarks

This paper has focused on reduction techniques for lazy learning algorithms based on generating new prototypes from the original training cases. Four new schemes have been introduced by using the concept of space partitioning. Two different division criteria have been analyzed, along with two distinct manners to create the new prototypes in each subset.

From the experiments, it seems that generating only one prototype for each subset gives lower accuracy than creating one for each class present in each partition. The split criterion affects accuracy, but much less than the method used for generating prototypes. Accordingly, IRSP1 and IRPS3 show the highest accuracy with a very small number of instances. Secondly, IRSP2 has the advantage that it allows to control the number of resulting cases and, it still obtains higher accuracy than Chen's algorithm. Finally, IRSP4 achieves the highest accuracy, although it also creates more prototypes than the other schemes.

References

1. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Machine Learning **6** (1991) 37-66.
2. Aha, D.W.: Tolerating noisy, irrelevant and novel attirbutes in instance-based learning algorithms. Int. Journal of Man-Machine Studies **36** (1992) 267-287.
3. Chang, C.-L.: Finding prototypes for nearest neighbor classifiers. IEEE Trans. on Computers **23** (1974) 1179-1184.
4. Chen, C.H., Józwik, A.: A sample set condensation algorithm for the class sensitive artificial neural network. Pattern Recognition Letters **17** (1996) 819-823.
5. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Trans. on Information Theory **13** (1967) 21-27.
6. Hart, P.: The condensed nearest neighbor rule. IEEE Trans. on Information Theory **14** (1968) 505-516.
7. Merz, C.J., Murphy, P.M.: UCI Repository of Machine Learning Databases. Univ. of California, Irvine. http://www.ics.uci.edu/mlearn (1998).
8. Wilson, D.L: Asymptotic properties of nearest neighbor rules using edited data sets. IEEE Trans. on Systems, Man and Cybernetics **2** (1972) 408-421.