Finding Small Consistent Subset for the Nearest Neighbor Classifier Based on Support Graphs

Milton García-Borroto[1,3], Yenny Villuendas-Rey[2], Jesús Ariel Carrasco-Ochoa[3], and José Fco. Martínez-Trinidad[3]

[1] Bioplantas Center, UNICA, Carretera a Morón km 9 ½, C. de Ávila, Cuba
mil@bioplantas.cu
http://www.bioplantas.cu
[2] Ciego de Ávila University UNICA, Carretera a Morón km 9 ½, C. de Ávila, Cuba
yennyv@bioplantas.cu
http://www.unica.cu
[3] National Institute of Astrophysics, Optics and Electronics, Puebla, México
{ariel,fmartine}@inaoep.mx
http://www.inaoep.mx

Abstract. Finding a minimal subset of objects that correctly classify the training set for the nearest neighbors classifier has been an active research area in Pattern Recognition and Machine Learning communities for decades. Although finding the Minimal Consistent Subset is not feasible in many real applications, several authors have proposed methods to find small consistent subsets. In this paper, we introduce a novel algorithm for this task, based on support graphs. Experiments over a wide range of repository databases show that our algorithm finds consistent subsets with lower cardinality than traditional methods.

Keywords: nearest neighbor, condensing, prototype selection, minimal consistent subset.

1 Introduction

Instance based learning has become one of the most used techniques in Machine Learning and Pattern Recognition. Among these techniques, the Nearest Neighbor (NN) classifier is one of the most popular supervised classifiers, due to its simplicity and high accuracy results. Two of its major advantages are that it does not assume any knowledge about data distribution, and its error is asymptotically bounded by twice the Bayes error [1].

However, in order to classify a new instance, NN computes its distance with all the objects in the training set. The computational cost of classification depends not only on the amount of objects in the training set, but also on the distance function, which in some domains, such as image processing, can be computationally expensive [2]. In addition, the storage cost depends on the amount of instances, and the number of features that describe them.

Since the introduction of the NN classifier, there is a constant research interest to find a reduced subset of instances with approximately the same classification power than the original set. Deleting redundant and irrelevant objects of the training set

E. Bayro-Corrochano and J.-O. Eklundh (Eds.): CIARP 2009, LNCS 5856, pp. 465–472, 2009.

seems to be one of the most successfully approach. By using a reduced set of objects, or condensed subset, it is possible to decrease both storage and classification cost.

There are subsets especially important in object reduction, named consistent subsets, which are those subsets that correctly classify the whole training set. Between them, those with minimum size, are the focus of many researches and papers. Although finding a consistent subset of minimum cardinality is a NP-complete problem [3], finding a small subset is a common goal in many researches [4-6] and it is still a challenge.

In this paper, we introduce a new method for obtaining an approximation to the Minimal Consistent Subset (MCS) for the Nearest Neighbor classifier. It iteratively selects the less useful object for keeping the consistency using the information on a support graph. A support graph includes all the information about which objects guarantee the correct classification of other objects. This new method frequently obtains an object subset with fewer objects than previous methods reported in the literature.

The paper is organized as follows: section two summarizes some of the previous works to find minimal consistent subsets, section three introduces our proposal, section four details the numerical results of the experiments and section five offers some conclusions.

2 Minimal Consistent Subset Selection

Finding consistent subsets of objects for the Nearest Neighbor classifier has been a problem of interest in Pattern Recognition since 1968 when Hart proposed the Condensed Nearest Neighbor (CNN) algorithm [4]. In this work, he introduced the concept of consistent subset, a subset of the objects that correctly classifies all samples in the training set. In 1991, Wilfong demonstrated that finding a consistent subset of minimum cardinality is a NP-complete problem [3]. However, since Hart´s work, there has been several attempts to find small consistent subsets.

Among the most cited methods for finding minimal consistent subsets are RNN [5] and MCS [6]. The Reduced Nearest Neighbor (RNN) consists in a post processing of the CNN algorithm. After applying CNN, RNN deletes an object if this deletion does not introduce any inconsistency. Gates [5] demonstrated that if the minimum consistent subset is a subset of the CNN result, then RNN method always finds it.

In MCS, every instance x gives a vote to each instance (of the same class) closer than the Nearest Unlike Neighbor (NUN) object. The NUN is the object from different class closest to x. The MCS algorithm iteratively constructs a consistent subset, adding the instance with most votes, and repeating the process until a consistent subset is found.

The Generalized CNN [7] is another method that extract a consistent subset using a stronger criterion for removing objects. Due to this, it always obtains supersets of CNN result, with larger size.

3 Condensation Based on Support Graphs

In this section, we introduce the CSESupport algorithm. It is a modification of the CSE algorithm [8], aiming to obtain a better approximation to the minimal consistent subset. It has the following changes:

- CSESupport uses a support graph instead of a nearest neighbor graph. A support graph is a directed graph, having arcs $x \rightarrow y$ if and only if the object y supports the object x. We say that y supports x if y is closer to x than x's NUN. This way if y belongs to the condensed subset, it guarantees the correct classification of x with a NN classifier. In a support graph, such vertexes with more inward arcs are the most important for condensing, because they support more objects.
- CSESupport makes multiple iterations. After each iteration, the algorithm extracts a smaller consistent subset than previous step, so the NUNs of some objects can differ. This way, the support graph can change, leading to potential new reductions.

Each iteration of CSESupport contains the following steps:

1. Support graph construction. The graph is constructed with respect to the objects in the current solution.
2. Add nodes with no successors (so they can not be supported by any other object) to the new solution if they are not assured, using the procedure *Move*. The *Move* procedure has three steps:
    a. Mark all antecessors of the node as assured. An assured node can be safely discarded from the training sample without affecting consistency.
    b. Delete all shared non-simultaneous deletion marks. These marks are a key point inherited from CSE to avoid inconsistencies in the result. They are added to the objects that support a deleted object, and if an object is the last one with such mark, it has to be included in the result. It is clear than the marks used by an object are different by those used by other object.
    c. Delete the node from the graph. Note than the assured nodes are not discarded in this step, because their inclusion in the result can assure more objects. This frequently led to smaller results.
3. Add nodes with the last non-simultaneous deletion mark to the new solution using Move.
4. Delete the less important node in the graph using the Discard procedure. A node is considered less important than other if it contains less inward arcs, because it is able to support less objects. Including in the result an object that supports more objects that other usually leads to smaller size consistent subsets. This criterion guides the CSESupport heuristic.
   The *Discard* procedure has three steps:
    a. If the node is not already assured, mark all its successors with a non-simultaneous deletion marks.
    b. Delete x of every non-simultaneous deletion marks in which it appears.
    c. Delete the node from the graph.
5. If the graph is not empty, go to step 2.

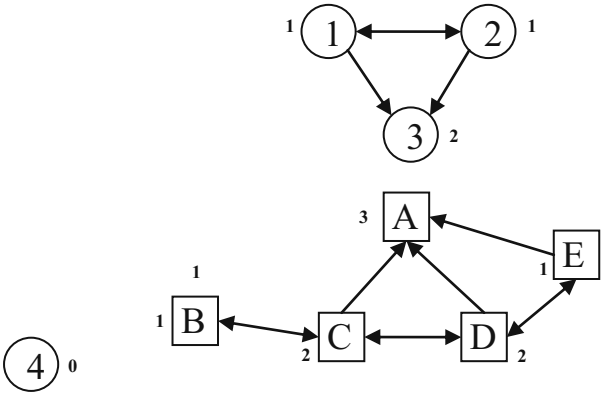


Fig. 1. Support graph with 9 object in 2 classes: circles and squares. The small numbers close to the shapes contains the number of inward arcs.

After an iteration the algorithm stops if the current solution is equal than previous solution. Otherwise, a new iteration is executed using the current solution to build the new support graph.

Now we will run an iteration of CSESupport with the objects in Fig. 1. In this example we build the support graph using the Euclidean distance between the centers of the shapes.

In step 2 nodes 3, 4 and A are *moved* to the current solution, because they have no successors. While *moving*, nodes 1, 2, C, D and E are assured. In step 4 the node 1 is *deleted*, but like it is assured no marking is necessary. In step 5 we loop to step 2. Step 2 can not be applied because the only node with no support (object 2) is already assured. In step 4 object 2 is *deleted.* Similarly object E is deleted in the next loop. Next loop node B is deleted in step 4, but like it is not assured the node C is marked with a non-simultaneous deletion mark. Next loop the step 3 *moves* node C to the result. Finally objects C and D are *deleted* without marking because they were previously assured. The resultant consistent subset is then {3,4,A,C}.

CSESupport, like CSE, is not able to deal with general k-NN classifiers, because the support graph only contains the information about the nearest neighbor.

4 Experimental Results

In order to test the behavior of CSESupport for finding small consistent subsets, we select RNN and MCS methods, which have been commonly used in experimental comparisons. Other methods, like CNN and GCNN, always produce larger results than RNN, so we dismiss them. For comparing the performance of the selected condensing methods, first we applied them over 2-D synthetic databases (section 4.1). We also carried out numerical experiments over a wide-range of databases from the UCI repository of Machine Learning [9] (section 4.2). The description of the repository databases appears in Tables 1.

The experiments with repository databases use 10-fold cross validation. Since NN classifier results depend on the function used for comparing objects, we use two

functions: HEOM and HVDM [10]. In order to determine the best method, we compute object retention for each one of them, and count how many times each method had the lower retention rate. For determining the reduction influence in classifier accuracy, we made a two-tailed T-Test with significance of 0.05, with respect to the method with lower classifier error, and compute how many times each method had the lower error, according to the T-Test.

Table 1. Description of repository databases

Database	non-num/ num feats	Objects	Database	non-num/ num feats	Objects
anneal	29/9	798	**hepatitis**	13/6	155
autos	10/16	205	**iris**	0/4	150
balance-scale	0/4	625	**labor**	6/8	57
breast-cancer	9/0	286	**lymph**	15/3	148
breast-w	0/9	299	**post-pat-data**	7/1	90
cmc	7/2	1473	**sonar**	0/60	208
colic	15/7	368	**tae**	2/3	151
credit-a	9/6	690	**trains**	29/4	10
cylinder-bands	20/20	512	**vehicle**	0/18	846
dermatology	1/33	366	**vote**	16/0	435
glass	0/10	214	**vowel**	3/9	990
heart-c	7/6	303	**zoo**	16/1	101
heart-h	7/6	294			

4.1 Results with Synthetic Databases

We generate two 2-D synthetic databases (see Fig. 1). Each database has three classes, represented by triangles, squares and circles, respectively. Database "Bananas and circle" is a modified version of the database "Bananas", used by Kuncheva [11]. Database "Venn´s diagram" has three overlapped classes, forming circles as in a Venn diagram. In gray, silver gray and white we represent the decision region of the classes squares, triangles and circles, respectively.

In figures 2-3, we show the results of each method over synthetic databases, using the Euclidean distance. The CSESupport method outperforms RNN in one and MCS in both synthetic databases.

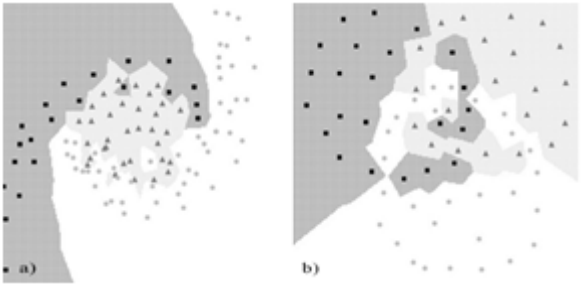


Fig. 2. a) "Bananas and circle", b) "Venn´s diagram"

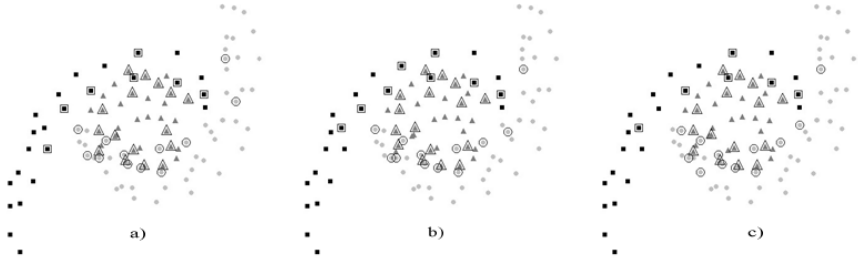


Fig. 3. Results of each method in database "Bananas and circle". a) RNN (35 obj.), b) MCS (36 obj.) and c) CSESupport (33 obj.).

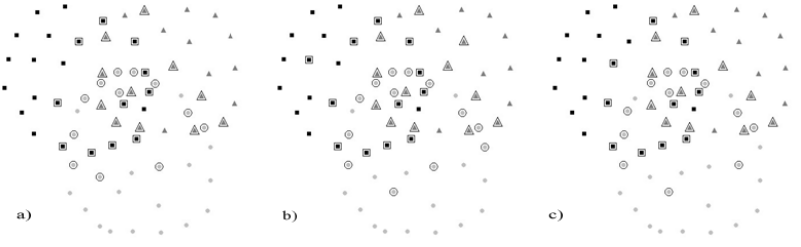
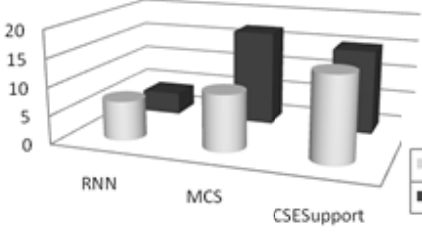


Fig. 4. Results of each method in database "Venn´s diagram". a) RNN (34 obj.), b) MCS (37 obj.) and c) CSESupport (34 obj.).

4.2 Results with Repository Databases

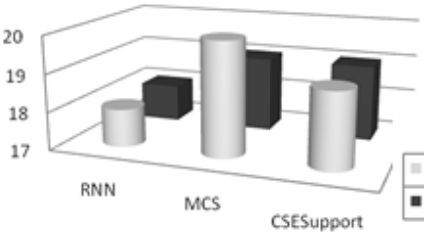
For repository databases, we averaged for each dissimilarity function the object retention results and counted how many times each method had the lowest object retention (see Fig. 5). According to our experiments, CSESupport achieved the best results, obtaining consistent subsets of lower cardinality more often than RNN and MCS.

These results can be explained because the quality of RNN result is strongly dependant on the CNN result. The CNN method is, on the other hand, strongly dependant on the order of the objects in the database. With respect to MCS, the difference is mainly due to the general search strategy: MCS inserts the most useful object, while CSESupport deletes the less important object. This way, like in other pattern recognition problems [12], sequential backward searches are more accurate than sequential forward searches.



	RNN	MCS	CSESupport
HEOM	7	10	15
HVDM	4	17	15

Fig. 5. Number of times each method achieved the lowest object retention



	RNN	MCS	CSESupport
HEOM	18	20	19
HVDM	18	19	19

Fig. 6. Number of times each method achieved the lowest classification error

We also compared the performance of the methods according to classification error. We made a two-tailed T-Test [13] with respect to the lowest error result, including the classifier trained with the original training set. We also count how many times each method had (statistically) the lowest error. The summary of the error classification results appears in Fig. 6. In general, all methods had similar results, maintaining low classification error in most of the tested databases.

5 Conclusions

Finding the minimal consistent subset is a computational intractable problem in many real life databases. Nevertheless, some authors have introduced heuristic proposals to find small consistent subsets. In this paper, we introduce a new algorithm for finding low cardinality consistent subsets for Nearest Neighbor classifiers, which usually find smaller subsets than state-of-the-art methods. The proposed method is based on support graphs, which provides valuable information in regions where objects from different classes are close together. In our experiments, the proposed condensing method (CSESupport) achieved the best results according to object retention, obtaining subsets with the lowest cardinality more often than MCS and RNN. As future work, we are working on including more information in the support graph to make CSESupport able to deal with k-NN classifiers.

Acknowledgments

The first author wants to thank to the National Institute of Astrophysic, Optics and Electronics for its support on this research. This work is partly supported by National Council of Science and Technology of México under grant 252752.

References

1. Cover, T., Hart, P.E.: Nearest Neighbor pattern classification. IEEE Trans. on Information Theory 13, 21–27 (1967)
2. Athitsos, V.: Learning embeddings for indexing, retrieval, and classification, with applications to object and shape recognition in image databases. Vol. Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, p. 156. Boston University (2006)

3. Wilfong, G.: Nearest neighbor problems. In: 7th Annual ACM Symposium on Computational Geometry, pp. 224–233 (1991)
4. Hart, P.E.: The condensed nearest neighbor rule. IEEE Trans. on Information Theory 14, 515–516 (1968)
5. Gates, G.W.: The reduced nearest neighbor rule. IEEE Transactions on Information Theory IT-18, 431–433 (1972)
6. Dasarathy, B.D.: Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design. IEEE Transactions on Systems, Man and Cybernetics 24, 511–517 (1994)
7. Chou, C.-H., Kuo, B.-H., Chang, F.: The Generalized Condensed Nearest Neighbor Rule as a Data Reduction Method. In: 18th International Conference on Pattern Recognition ICPR 2006, Tampa, USA. IEEE, Los Alamitos (2006)
8. García-Borroto, M., Ruiz-Shulcloper, J.: Selecting Prototypes in Mixed Incomplete Data. In: Sanfeliu, A., Cortés, M.L. (eds.) CIARP 2005. LNCS, vol. 3773, pp. 450–459. Springer, Heidelberg (2005)
9. Merz, C.J., Murphy, P.M.: UCI Repository of Machine Learning Databases. University of California at Irvine, Department of Information and Computer Science, Irvine (1998)
10. Wilson, R.D., Martinez, T.R.: Improved Heterogeneous Distance Functions. Journal of Artificial Intelligence Research 6, 1–34 (1997)
11. Kuncheva, L.I.: Combining pattern classifiers: methods and algorithms. Wiley-Interscience, Hoboken (2004)
12. Pudil, P., Novovicova, F.J., Kittler, J.: Floating search methods in feature selection. Pattern Recognit. Lett. 15, 1119–1125 (1993)
13. Dietterich, T.G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms, vol. 10, pp. 1895–1923. MIT Press, Cambridge (1998)