A Modular Reduction Method for *k*-NN Algorithm with Self-Recombination Learning

Hai Zhao and Bao-Liang Lu*

Department of Computer Science and Engineering, Shanghai Jiao Tong University,
800 Dong Chuan Rd., Shanghai 200240, China
bllu@sjtu.edu.cn

Abstract. A difficulty faced by existing reduction techniques for k-NN algorithm is to require loading the whole training data set. Therefore, these approaches often become inefficient when they are used for solving large-scale problems. To overcome this deficiency, we propose a new method for reducing samples for k-NN algorithm. The basic idea behind the proposed method is a self-recombination learning strategy, which is originally designed for combining classifiers to speed up response time by reducing the number of base classifiers to be checked and improve the generalization performance by rearranging the order of training samples. Experimental results on several benchmark problems indicate that the proposed method is valid and efficient.

1 Introduction

Given a training set of previously labeled samples and an unknown sample x, the k-nearest neighbor (k-NN) algorithm assigns the most frequently represented class-label among the k closest prototypes to x. The k-NN algorithm and its derivatives have been shown to perform well for pattern classification in many domains over the last 40 years. In theory, the k-NN algorithm is asymptotically optimal in the Bayesian sense [1]. In other words, k-NN performs as well as any other classifier, provided that there is an arbitrarily large number of (representative) prototypes available and the volume of the k-neighborhood of x is arbitrarily close to zero for all x.

Basically, the k-NN algorithm needs the entire training set as the representative set. Therefore, the space requirement of storing the complete set and the high computational cost for the evaluation of new samples are often criticized. The k-NN algorithm also shows sensitivity to outliers, i.e. noisy or even erroneously labeled prototypes. To overcome these drawbacks, various techniques have been developed. Two main types of algorithms can be identified: prototype

* To whome correspondence should be addressed. This work was supported by the National Natural Science Foundation of China under the grants NSFC 60375022 and NSFC 60473040.

generation and prototype selection. The first group focuses on merging the initial prototypes into a small set of prototypes, so that the performance of the k-NN algorithm is optimized. Examples of such techniques are k-means algorithm [2] and learning vector quantization algorithm [3]. The second group aims at the reduction of the initial training set and/or increasing the accuracy of the NN prediction. Various editing and condensing methods were developed, such as condensing algorithm [4] and editing algorithm [5]. Wilson *et al.* recently gave a good review for all these reduction techniques for the k-NN algorithm [6].

All of these existing reduction algorithms, however, still need to store the whole training set firstly to meet the requirement of fast processing. The reason is that all of them are actual k-NN based self-test of the training set. Therefore, if there is any no parallel or modular techniques introduced to the k-NN algorithm, those existing k-NN reduction techniques can not efficiently perform reduction of a large-scale data set.

A modular k-NN algorithm has been proposed in our previous work [7]. Of course, this modular k-NN algorithm is mainly suitable to dealing with very large-scale data sets, and a user can not divide a training data set into too many subsets in order to maintain a stable combining accuracy. Otherwise, some assistant techniques, such as clustering, should be introduced to help the partition of the training data set [8][9]. However, all clustering procedures also require additional computing time and the same storage space unless there is a parallel or modular version of clustering algorithm. In one word, unless we can directly solve the reduction problem for the k-NN algorithm or there exists a method which can carry out both partition and reduction at the same time, we will face many other concomitant problems.

In this paper, s self-recombination learning scheme based reduction method is proposed for overcoming above difficulties. The proposed method is based on combining classifier from the quadratic decomposition of two-class task. It is also a framework of parallel or modular self-test based on the k-NN algorithm.

2 Combining Classifier

Since any multi-class classification problem can be solved by using two-class classification techniques according to well known one-against-rest and one-against-one task decomposition methods. Especially, the one-against-one task decomposition method has been an innate parallel and modular processing method for binary classifier modules. Thus, we will only concern with two-class combining classifier in the remained part of this paper.

We firstly discuss the decomposition of a two-class classification task. Let $\mathcal{X}^+$ and $\mathcal{X}^-$ be the given positive and negative training data sets, respectively, for a two-class classification task T,

$$\mathcal{X}^+ = \{(\mathbf{x}_i^+, +1\}_{i=1}^{l^+}, \quad \mathcal{X}^- = \{(\mathbf{x}_i^-, -1)\}_{i=1}^{l^-} \tag{1}$$

where $\mathbf{x}_i^+ \in \mathbf{R}^n$ and $\mathbf{x}_i^- \in \mathbf{R}^n$ are the input vectors, and l^+ and l^- denote the numbers of positive training samples and the number of negative training samples, respectively.

$\mathcal{X}^+$ and $\mathcal{X}^-$ can be partitioned into N^+ and N^- subsets respectively,

$$\begin{aligned} \mathcal{X}_j^+ &= \{(\mathbf{x}_i^{+j}, +1)\}_{i=1}^{l_j^+}, \quad j = 1, \ldots, N^+ \\ \mathcal{X}_j^- &= \{(\mathbf{x}_i^{-j}, -1)\}_{i=1}^{l_j^-}, \quad j = 1, \ldots, N^- \end{aligned} \tag{2}$$

where $\cup_{j=1}^{N^+} \mathcal{X}_j^+ = \mathcal{X}^+$, $1 \leq N^+ \leq l^+$, and $\cup_{j=1}^{N^-} \mathcal{X}_j^- = \mathcal{X}^-$, $1 \leq N^- \leq l^-$.

Matching all these subsets, $\mathcal{X}_i^+$ and $\mathcal{X}_j^-$, where $i = 1, \ldots, N^+$ and $j = 1, \ldots, N^-$, can yield $N^+ \times N^-$ relatively smaller and more balanced (if needed) two-class subsets pairs. Each of them can be expressed as

$$T^{(i,j)} = \mathcal{X}_i^+ \bigcup \mathcal{X}_j^-, i = 1, \ldots, N^+, j = 1, \ldots, N^- \tag{3}$$

Assign a k-NN classifier as a base classifier to learn from each subset $T^{(i,j)}$. In the learning phase, all of the k-NN base classifiers are independent from each other and can efficiently work in a parallel way. We also denote the corresponding k-NN base classifier as $T^{(i,j)}$, if there is not any misunderstanding.

For the convenient, these base classifiers, $T^{(i,j)}, j = 1, \cdots, N^-$, are defined as 'positive group' and i is defined as its group label, and those base classifiers, $T^{(i,j)}, i = 1, \cdots, N^+$, are defined as 'negative group' and j is defined as its group label.

Intuitively, if all classifiers in a positive group support the decision output of positive class for an unknown input sample (notice that this is a very strong condition), then to assign this unknown sample the positive class label is very natural. It is the same for a negative sample and its related negative group. We call a positive group whose all member base classifiers support the positive class as a 'positive winning group' and a negative group whose all member base classifiers support the negative class as a 'negative winning group'.

It should be noted that it is impossible that positive and negative winning groups exist at the same time, because these two groups must share one same base classifier and this classifier can not output two different classification results at the same time.

To finish a combination without any preference, we define the combination strategy as follows. The combining output of the positive class is determined by the condition that no negative winning groups exist, and the combining output of the negative class is determined by the condition that no positive winning groups exist. Sometimes, neither positive nor negative winning group exits in application, that is the reason why we make the definitions with the negatory conditions.

In order to effectively handle the case that no winning group exists and realize an efficient combination, the following symmetrical selection algorithm is given as a practical realization of the above combining strategy.

1) Set the initial labels, $i = 1$ and $j = 1$.
2) Repeat the following operations:
   (a) Check the base classifier $T^{(ij)}$.
   (b) If $T^{(ij)}$ supports the positive class, then $j = j + 1$.
   (c) If $T^{(ij)}$ supports the negative class, then $i = i + 1$.
   (d) If $i = 1 + N^+$, then the combining output is negative class and the algorithm ends here.
   (e) If $j = 1 + N^-$, then the combining output is positive class and the algorithm ends here.

We now show that the number of checked base classifiers in symmetrical selection algorithm for each input sample will be never larger than $N^+ + N^- - 1$. As an intuitive expression, we imagine that all outputs of base classifiers are represented in a binary matrix with N^- columns and N^+ rows. It is obvious that the algorithm is equally a search procedure in the matrix, in which the start point is the top-left corner and the end point is near the bottom-right corner. Each checking for a base classifier means one row or one column in the matrix must be excluded. The search direction will turn to the next row or column without any backtracking after each checking. Consider the maximum-length search from the top-left corner to the bottom-right corner in the matrix. This maximum-length search will cover $N^+ + N^- - 1$ elements. That is, the number of checked base classifiers in symmetrical selection for an input sample will not be larger than $N^+ + N^- - 1$, which is much less than the number of all base classifiers, $N^+ \times N^-$.

The theoretical analysis in our existing study has actually given a probability model for symmetrical selection based combination through the performance estimation of two-class combining classifier [10].

3 Reduction with Self-Recombination Learning

Self-recombination learning (SRL) originates such a simple idea. Consider the searching procedure of symmetrical selection algorithm in the outputs matrix of all base classifiers, the start point is the base classifier in the top-left corner, and the end point will be located in the most right column or the most below rows. We may notice if a base classifier is nearer to the bottom-right point of the output matrix, it will have the less chance to be checked and the less capability to determine the final combination output. Thus, we may obtain the higher combining accuracy by arranging those base classifiers with higher error rate to the more right columns and the more below rows. Finally, we may remove the learning of those base classifiers from 'bad' samples to realize the sample reduction.

The proposed reduction algorithm with self-recombination learning can be described as follows.

1) Specify the volumes of all subsets.

2) Set the maximum turns of self-learning, M, and the counter of the self-learning times, $t = 1$.
3) While $t < M$, do
   (a) Extract specified number of training samples in order from either training set to yield all training subsets of individual class.
   (b) Record the classification outputs of all k-NN base classifiers for all samples in the training set.
   (c) Record the combining outputs of all samples of the training set according to the symmetrical selection combination.
   (d) Put all incorrectly classified samples in the rear part of the set for both positive and negative training sets.
4) Remove all k-NN base classifiers learning from incorrect classified samples.

There are still two potential problems remained in the above self-learning scheme. One is how to determine the stop criterion of SRL. We experimentally suggest that a stable training accuracy can be taken as such a criterion. The other is how to rearrange the correctly classified samples in proper order. Always, we can finish these two tasks in one step, moving the correctly or incorrectly classified samples to the corresponding parts and rearranging the order of the samples in either part. We call such procedure as moving and rearranging algorithm (MRA). Assume the number of all samples is N_{Max}, a suitable MRA for the less exchange times and the relatively stable learning can be described as follows.

1) Set the serial number of the sample to be handled, $i = 1$.
2) While $i <= N_{Max}$, do
   (a) If the sample i is correctly classified, then $i = i + 1$. Otherwise, do
      i. Set the counter, $C = 0$, and the serial number, $j = i + 1$.
      ii. If the sample j is correctly classified, then exchange samples i and j and goto a), Otherwise, let $j = j + 1$ and $C = C + 1$.
      iii. If $C = N_{Max} - j$, then the algorithm ends here. Otherwise, go to ii..

4 Experiments

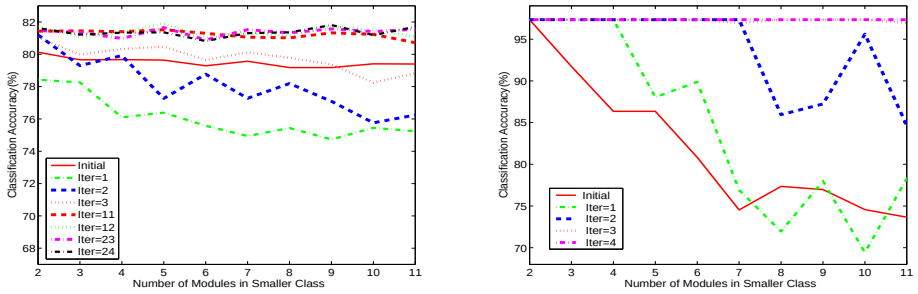
Two data sets as shown in Table 1 from UCI Repository [11] and STATLOG benchmark repository [12] are chosen for this study. To carry out decomposition of the training data set, the smaller class is divided into some equal size subsets, and the larger class is also equally divided into some subsets with the same size as the subsets of smaller classes. In k-NN algorithm, the optional values of k are set to $7, 13, 19$, and 25, respectively.

Comparisons of classification accuracy after reduction through multi-turn self-recombination learning on every data sets are shown in Figs. 1. The experimental results show that the more stable and higher combining accuracy appears as the iteration of self-recombination learning increases.

To show the classification accuracy of reduction is superior to the classification accuracy of using the whole samples, we give a comparison study on two

Table 1. Distributions of the samples in each class on all data sets

Data Set	Number of Training Samples			Number of Test Samples			#Input
	Total	Positive	Negative	Total	Positive	Negative	
Breast cancer	20000	14118	5882	7700	5482	2218	9
Census income	20107	15102	5005	10055	7552	2503	14



(a) (b)

Fig. 1. Classification accuracy after reduction through multi-turn self-recombination learning and different scale partition. (a) Census income data set, where $k = 7$ and (b) Breast Cancer data set, where $k = 25$.

cases. The experimental results shown in Fig. 2 verifies the fact that the reduction is effective on Census income data set. Actually, the comparison results are the same for the other values of k on Census income and other data sets. However, the comparison results for the other two data sets are very close, since their original classification accuracies have been very high.

Comparisons of test accuracy with reduction and training accuracy (also is the storage percentage after reduction) after multi-turn self recombination learning on every data sets are shown in Fig. 3. The experimental results show the consistent trend for the training accuracy and the test accuracy, which suggest that a stable training accuracy does be taken as the stop criterion of self learning.

5 Related Work

We use a 'quadratic' decomposition of two-class training data set, that is, the number of produced subsets or base classifiers are the product of each number of single-class subsets. Min-max modular (M^3) classifier [13] is also a kind of quadratic decomposition based combining classifier. The difference between the proposed combining classifier and M^3 classifier is in the combination strategy. Min-max combination in M^3 classifier needs to check $O(N^+ \times N^-)$ base classifiers for an input sample, while symmetrical selection combination only needs $N^+ + N^- - 1$ at most. In addition, the min-max combination prefers the output of negative class label for the reason that the min-max combining procedure is

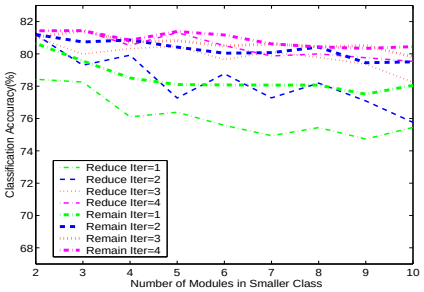


Fig. 2. Comparison of test accuracy with reduction and without reduction on Census income data set, $k = 7$.

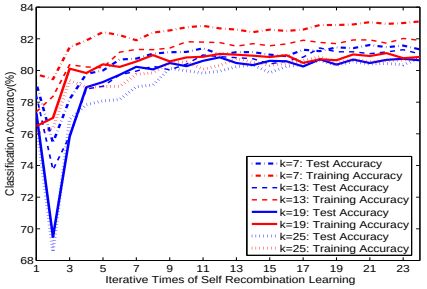


Fig. 3. Comparison of training accuracy (storage percentage) and test accuracy in Census income data set, the smaller class is divided into 8 parts.

asymmetrical for two different classes. As for the combining accuracy, our experimental results show two combination strategies are very similar at most cases, and symmetrical selection combination is some better at some unbalanced cases.

Surely, the quadratic decomposition is not a common task decomposition method. Most combining classifiers are based on a 'linear' decomposition of training data set, that is, the whole training data set is divided into some parts [14]. This kind of decomposition produces less subsets and base classifiers. However, it will lose some flexibility to control ratio of the numbers of samples among every classes, while the quadratic decomposition is just on the contrary. Of course, linear decomposition can not provide the mechanism of the reduction method developed in this paper.

Some recent works are also concerned with modular reduction [15]. The difference between our work and the existing approach is that it only considers an extreme case that each subset contains only two different samples and this approach is only suitable for 1-NN reduction, while our scheme is more general. In the extreme case, the combining classifier in [15] equally performs like a 1-NN classifier. Therefore, it is not strange that self-recombination learning based reduction in such case is equal to edited nearest neighbor rule of Wilson [16].

6 Conclusions

A modular and parallel reduction method with self-recombination learning for k-NN algorithm has been proposed in this paper. One of the most important features of the proposed method is that the adjustment of partitioned training data set and the reduction task can be done at the same time, while the scales of all decomposed training subsets can always be kept unchanged. Thus, the proposed method is also an effective modular k-NN algorithm. As for the future work, we expect for a more quick and stable moving and rearrangement algorithm to improve self-recombination learning. More effective stop criterion of SRL is required, too.

References

1. Dasarathy, B. V. (ed.): Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques. Los Alamitos, CA: IEEE Computer Society Press. (1991)
2. MacQueen, J. B.: Some Methods for Classification and Analysis of Multi Variate Observations. Berkeley Symposium on Mathematical Statistics and Probability (1967) 281-297
3. Kohonen, T.: Self-Organizing Maps. Springer, Germany. (1995)
4. Hart, P. E.: The Condensed Nearest Neighbor Rule. IEEE Trans. on Information Theory 14(5) (1967) 515-516
5. Devijver, P. A. & Kittler, J.: Pattern Recognition: A Statistical Approach. Prentice/HallLondon. (1982)
6. Wilson, D. R. & Martinez, T. R.: Reduction Techniques for Instance-Based Learning Algorithms. Machine Learning 38(3) (2000) 257-286
7. Zhao, H. & Lu, B. L.: A Modular k-Nearest Neighbor Classification Method for Massively Parallel Text Categorization. In: Zhang, J., He, J.-H., Fu, Y. (eds.): Computational and Information Science: First International Symposium, CIS 2004. Lecture Notes in Computer Science, Vol. 3314. Springer-Verlag, Berlin Heidelberg New York (2004) 867 - 872
8. Giacinto, G. & Roli, F.: Automatic Design of Multiple Classi-fier Systems by Unsupervised Learning. Proceedings of Machine Learning and Data Mining in Pattern Recognition, First Interna-tional Workshop, MLDM'99, Leipzig, Germany (1999) 131-143
9. Frosyniotis, D., Stafylopatis A. & Likas, A.: A Divide-and-conquer Method for Multi-net Classifiers. Pattern Anal Applic 6 (2003) 32-40
10. Zhao, H.: A Study on Min-max Modular Classifier (in Chinese). Doctoral dissertation of Shanghai Jiao Tong University (2005)
11. Blake, C. L. & Merz, C. J.: UCI Repository of Machine Learning Data-bases [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Com-puter Science (1998)
12. Ratsch, G., Onoda T. & Muller, K.: Soft Margins for AdaBoost. Machine Learning (2000) 1-35
13. Lu, B. L. & Ito, M.: Task Decomposition and Module Combination Based on Class Relations: a Modular Neural Network for Pattern Classification. IEEE Transactions on Neural Networks 10 (1999) 1244-1256
14. Chawla, N. V., Hall, L. O., Bowyer, K. W. & Kegelmeyer, K. P.: Learning Ensembles from Bites: A Scalable and Accurate Approach. Journal of Machine Learning Research 5 (2004) 421-451
15. Yang, Y. & Lu, B. L.: Structure Pruning Strategies for Min-Max Modular Network. In: Wang, J., Liao, X. and Yi Z. (eds.): Advances in Neural Networks – ISNN 2005: Second International Symposium on Neural Networks. Lecture Notes in Computer Science, Vol. 3496. Springer-Verlag, Berlin Heidelberg New York (2005) 646 - 651
16. Wilson, D. L.: Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Transactions on Systems, Man, and Cybernetics 2-3 (1972) 408-421