A Modular k-Nearest Neighbor Classification Method for Massively Parallel Text Categorization

Hai Zhao and Bao-Liang Lu

Department of Computer Science and Engineering, Shanghai Jiao Tong University, 1954 Huashan Rd., Shanghai 200030, China
{zhaohai, blu}@cs.sjtu.edu.cn

Abstract. This paper presents a Min-Max modular k-nearest neighbor (M^3-k-NN) classification method for massively parallel text categorization. The basic idea behind the method is to decompose a large-scale text categorization problem into a number of smaller two-class subproblems and combine all of the individual modular k-NN classifiers trained on the smaller two-class subproblems into an M^3-k-NN classifier. Our experiments in text categorization demonstrate that M^3-k-NN is much faster than conventional k-NN, and meanwhile the classification accuracy of M^3-k-NN is slightly better than that of the conventional k-NN. In practical, M^3-k-NN has intimate relationship with high order k-NN algorithm; therefore, in theoretical sense, the reliability of M^3-k-NN has been supported to some extend.

1 Introduction

In all categorization algorithms that based on vector space model, k-nearest neighbor (k-NN) method is considered as taking on the best performance in some sense. The k-NN method is very concise and easy to use, which simply estimates the test sample's class as the class which most supported in its k nearest neighbor training samples. Because of its advantage, k-NN method has been widely used in text categorization. Many researchers focus their endeavor on the improvement of text categorization ([2], [5], [7]). Commonly, these researches adopt additional information for improving the ultimate classification result.

However, the parallel techniques have been developed for traditional categorization for high-end application. Our research goal is to develop faster and more efficient methods for large-scale text categorization by direct modular classification without reducing the precision of the classification. Therefore we introduce a modular k-NN algorithm, M^3-k-NN.

In fact, many efficient methods have been proposed in the past decade in task decomposition and combination of modular classifiers ([1], [3], [6]), and the effectiveness of these methods have been studied and proved. But, thus far, no divide-and-conquer method has been put forward for k-NN algorithm.

J. Zhang, J.-H. He, and Y. Fu (Eds.): CIS 2004, LNCS 3314, pp. 867–872, 2004.

The basic ideal of our method is to adopt the k-NN algorithm in the combination of Min-Max Modular (M^3) Neural Network Model ([6]) and Round Robin Rule (R^3) Learning algorithm ([3]). Our experiments demonstrate that under parallel pattern, R^3-k-NN or R^3-M^3-k-NN has the similar recognition precision to the traditional k-NN and sometime even better. Also, the flexibility of this modular combination method can be found obviously for large-scale parallel processing by using grid or large-scale cluster systems.

2 Algorithm Description

We introduce the modular algorithm, M^3-k-NN and one of its revised versions, R^3-M^3-k-NN. The whole algorithm has been divided into two phases.

In the first phase, we simply use one-against-one task decomposition method. Suppose the number of classes for original classification problem is m, then we will have $m(m-1)/2$ independent base k-NN classifiers. The training samples of each base k-NN classifier come from two different classes. To combine the results, we consider two module combination policies as the following:

P_1 M^3-k-NN voting: In all base k-NN classifiers, if the outputs of $m-1$ base k-NN classifiers support the same class, then the classification result is just this class.

P_2 R^3-M^3-k-NN voting: The result is the class that supported by most base k-NN classifiers.

The policies P_1 and P_2 are exactly the same voting procedures used in [6] and [3], respectively. And policy P_2 is also one of voting strategies for multi-class support vector machine.

In the second phase, we further divide a two-class subproblem from the first phase into a number of smaller two-class subproblems. The decomposition approach is to decompose two training sample sets pair in a two-class classification problem into m_1 and m_2 smaller sample sets, respectively. Thus, we get $m_1 m_2$ smaller two-class classification sub-problems. Let the output coding of class ID is 1 and 0. We define all base classifiers learning from the same training set of class 1 as 'a group'. Then M^3 combination results of all base classifiers includes two stages: Firstly, combination rule ***Min*** is applied in each 'group' to produce a group output. Secondly, all outputs of every groups are applied by combination rule ***Max*** to produce the final combination output.

3 Experiments

We use Yomiuri News Corpus for this study. There are 2,190,512 documents in the full collections from the years 1987 to 2001. Two subsets of the corpus are used in the experiments, which belong to 10 classes. In the first subset, the number of training and test samples is 5,865 and 2,420, respectively. In the second subset, the number of training and test samples is 37,938 and 16,234, respectively.

A χ^2 statistical method ([7]) was used for preprocessing the documents after the morphological analysis was done with ChaSen. The number of features is 500.

The goal of the experiments is to compare recognition accuracy between k-NN and M^3-k-NN under the value of k varying. And we thus do not apply any optimization technology to increase the absolute accuracy. All simulations were performed on a PC with 2.4GHz Pentium 4 CPU and 512MB RAM.

3.1 M^3-k-NN, R^3-M^3-k-NN and k-NN: Accuracy in Training Set 1

We divide training sample sets from each class at the same size 2000 for some modules, if there are still remained samples in a class whose number is less than 2000, we put them into a new module. The value of k varies from 3 to 20. The experiment results are shown in Fig.1.

Obviously, while the value of k increases, the incorrect accuracy of both M^3-k-NN and R^3-M^3-k-NN increases after decreasing, which changes more rapidly than k-NN does. In fact, both M^3-k-NN and R^3-M^3-k-NN get to their peak accuracy earlier than k-NN does. Within the whole interval while the value of k changes, M^3-k-NN, R^3-M^3-k-NN or k-NN performs in a similar manner.

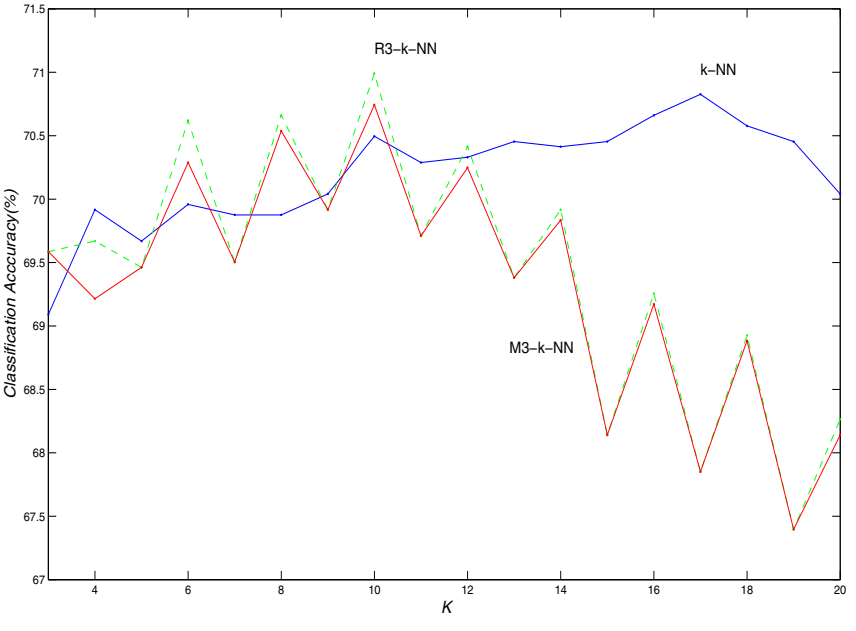

Fig. 1. M^3-k-NN, R^3-M^3-k-NN and k-NN: Accuracy in Training Set 1

3.2 M^3-k-NN and k-NN: Accuracy in Training Set 2

In the following experiment training set 2 is used. Here, we only compare k-NN with M^3-k-NN. The result is shown in Fig.2, from which we can see that M^3-k-NN gets to its peak accuracy earlier than k-NN does, too.

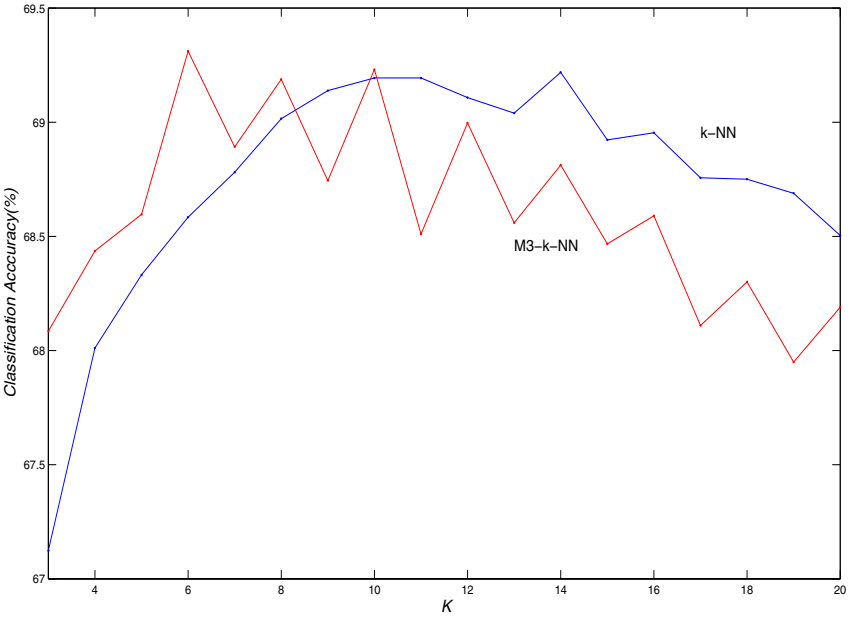

Fig. 2. M^3-k-NN and k-NN: Accuracy in Training Set 2

3.3 Comparison of Running Time

Comparison of classification time between two algorithms at each different value of k is shown in Table 1. All experiments run in a PC with Windows XP OS. Thus we estimate M^3-k-NN's parallel processing performance by dividing the total running time of M^3-k-NN by the number of modules. It is easy to see that the classification time used by M^3-k-NN in a parallel model is much less than that of traditional k-NN.

4 Theoretical Analysis of the Algorithm

Parts of our algorithm have been studied in [1] and [3], which shows that the load of a modular algorithm is lighter than the same one without any modularization in the phase 1 of the algorithm described in Section 2. This is an outstanding merit of all one-against-one modularization methods.

For an m-class problem, suppose the number of all training samples is T. If one visiting for a training sample in k-NN algorithm is taken as a basic unit of time complexity, then the time complexity of k-NN algorithm will be about kT.

As to the first phase of M^3-k-NN, the time complexity of each k-NN base classifier is near to $2kT/m$. Although the whole time complexity is $(m-1)kT$ under serial processing, the time complexity under parallel processing is only $2kT/m$. This means that the processing time of M^3-k-NN will be decreased if only there is a synchronous increase of both the number of classes and the

Table 1. Comparison of classification time between k-NN and M^3-k-NN

k	Training-Set 1 (ms)			Training-Set 2 (ms)		
	k-NN	M^3-k-NN	single M^3-k-NN module	k-NN	M^3-k-NN	single M^3-k-NN module
3	13672	68569	1524	564219	2781059	61801
5	14406	72428	1610	559344	2816960	62599
7	15094	75881	1686	573891	2873521	63856
9	14797	74217	1649	553125	2912828	64730
11	14828	74507	1656	573625	3348766	74417
13	15438	78597	1747	570875	3207375	71275
15	15407	77900	1731	584172	3737203	83049
17	15188	75951	1688	598421	3776235	83916
19	15766	79188	1760	610797	3824140	84981

number of processing units. If the second phase of the algorithm is considered, the parallel processing time complexity will be decreased more. In the extreme case, the value can be $2kT/(mm_1m_2)$.

Now we will give a simple explanation for the similar accuracy between M^3-k-NN and k-NN under the first task decomposition. There will be $m(m-1)/2$ sub-tasks in M^3-k-NN for an original m-class classification problem. A base k-NN classifier learning from samples of class i, j, while $1 \leq i, j \leq m$, or its corresponding output is denoted by X_{ij}.

We consider the composition of k nearest neighbors of a test sample while the original k-NN algorithm that is applied in all m-class training samples. We say, the k nearest neighbors must come from the union set of k nearest neighbors set of each base k-NN classifier's corresponding test sample in M^3-k-NN. Otherwise, suppose there is a sample from class j' outside the set at least, that is to say, it is in k nearest neighbors of m-class training set. Consider there is unique class j' in m classes. If we limit the class upon class i and j' while searching k nearest neighbors, the sample must come from either class i or class j', which contradict our initial suppose. Thus, k nearest neighbors from m-class is the subset of the union set of each base k-NN classifier's corresponding test sample's k nearest neighbors set in M^3-k-NN. This result suggests that there is a fixed ratio between M^3-k-NN's k voting sets and k-NN's. Especially, let k be equal to 1, thus, all samples from M^3-k-NN's voting set will be in the same class. Then, corresponding nearest neighbors in m classes must give the output with the same class number. Namely, M^3-1-NN is equal to 1-NN.

The reason that M^3-k-NN gets to its peak accuracy faster than k-NN does is that less samples are processed in each module for M^3-k-NN, then increasing of the value of k will have more notable effects on M^3-k-NN than k-NN.

However, in [4], another kind of k-NN classifiers combination has been considered, too. The main ideal of that paper is that k-NN classifiers' combinations are made under different feature combinations, and all k-NN classifiers are applied to the whole training text sets, while our method is to cut the whole training sets into different parts and then to perform individual k-NN classification.

5 Conclusions

We present a modular k-NN method, M^3-k-NN and its revised version R^3-M^3-k-NN in this paper, which are applied to large-scale text categorization. Our experiments show that M^3-k-NN admits a flexible k-NN classifiers modularization to realize a complete parallel processing, which will not reduce classification accuracy at the same time. In fact, as the value of k increasing, M^3-k-NN gets to its peak classification accuracy faster than k-NN does. In addition, our analysis suggests that M^3-k-NN algorithm concerns with a high order k-NN algorithm, which makes some theoretical guarantee for M^3-k-NN's reliability.

Acknowledgements

The authors would like to thank Junjie Zhang for his programming. This research is partially supported by the National Natural Science Foundation of China under Grant No. 60375022.

References

1. Allwein, E.L., Schapire, R. E., Singer, Y.: Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. Journal of Machine Learning Research, Vol.1 (2000) 113-141
2. Chiang, J.-H., Chen, Y.-C.: Hierarchical Fuzzy-KNN Networks for News Documents Categorization. The 10th IEEE International Conference on Fuzzy Systems, Vol.2 (2001) 720-723
3. Frnkranz,J.: Round Robin Classification. The Journal of Machine Learning Research, Vol.2 (2002) 721-747
4. Han, H., Yand J.Y., Hu, Z.S.: Combination of KNN Classifiers Based on Multiple Correspondence Analysis. Information and Control, Vol.28 (1999) 350-356
5. Lim, H.S.: An Improved KNN Learning Based Korean Text Classifier with Heuristic Information. ICONIP '02, Vol.2 (2002) 731-735
6. 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, Vol.10 (1999) 1244-1256
7. Yang, Y., Petersen, J.P.: A Comparative Study on Feature Selection in Text Categorization. ICML'97, (1997) 412-420.