Improved Hierarchical SVMs for Large-scale Hierarchical Text Classification Challenge*

Xiao-Lin Wang[1], Bao-Liang Lu[1,2]**

[1]Center for Brain-Like Computing and Machine Intelligence
Department of Computer Science and Engineering
[2]MOE-Microsoft Key Laboratory for Intelligent Computing and Intelligent Systems
Shanghai Jiao Tong University
800 Dong Chuan Road, Shanghai 200240, China
{arthur_general,bllu}@sjtu.edu.cn

Abstract. This paper describes an improved decision tree of multiclass SVM classifiers, with which we participate in the basic task and the cheap task of large-scale hierarchical text classification challenge. The structure of the decision tree is a simplified version of the hierarchy of the classes, and the optimal learning parameter of each node has been automatically searched. For two tasks of the challenge, the time cost of our system on a normal person computer is reasonable, and its performance is competitive.

We participate in large-scale hierarchical text classfication with an system of improved hierarchical SVMs. This paper first describes our system in Sec. 1, and then discusses the computational complexity in Sec. 2. After that it presents the accuracies, hardware settings and running time of our system in the tasks in Sec. 3. Finally it gives our conclusions and future works in Sec. 4.

1 Algorithmic Description

The work flow of our system is that it first performs TFIDF feature indexing on the samples [1], then simplifies the hierarchy of classes, then searches for the optimal parameter of SVM on each node by the training set and the validation set, then trains a decision tree of SVMs, at last uses it to classify test samples. In the formal task (the large dataset), the training is performed on both the training set and the validation set, which is officially allowed. The rest of this section describes two key issues of simplifying hierarchies and using multiclass SVMs as base classifiers.

* This work was partially supported by the National Natural Science Foundation of China (Grant No. 60903119, Grant No. 60773090 and Grant No. 90820018), the National Basic Research Program of China (Grant No. 2009CB320901), and the National High-Tech Research Program of China (Grant No.2008AA02Z315).

** Corresponding author

1.1 Simplifying Hierarchies

Simplifying the origin hierarchy of classes can raise the accuracy of the classifying system, which has been proved by our experiments.

The origin hierarchy is not proper to organize SVMs for two reasons. First, the number of the children or the training samples from a parent node may be too small to train a good SVM classifier, which both costs extra time and reduces accuracies. Second, the path from the root to some leafs may be too long to prevent the system from classifying correspondent samples correctly.

Simplifying the hierarchy of classes is done by emerging the small nodes, that is, the nodes with small number of children. We only emerge the nodes of the same depth in order to reserve the reasonability of the origin hierarchy. The algorithm is that, walk down from a parent node by in-order traversing till a level that the sum of descendants exceeds a threshold, then take these descendants as the direct children of the parent; after that continue the algorithm from these nodes.

The simplification improves the accuracy, but it increases computation complexity of a single node though reduces the global complexity. What is worse, the emerging may produce a node on which training a SVM exceeds the ability of a personal computer, which make the system unpractical. So there is a trade-off between the accuracy and the individual computation complexity. On both the basic task and the cheap task, we finally adopt the same simplified hierarchy, considering our 64bit computer with 8G physical memory. This hierarchy has only two levels, that is, it takes the 311 grandchildren (nodes at the second level) of the root as its direct children, and then takes all the leave nodes as the children of these 311 nodes.

1.2 Base Classifiers of Multiclass SVM

Joachims' implement of multiclass SVM is taken as the base classifiers of our system [2]. It is a derivation of SVM^{struct}, a package that can deal with structured classification problems. The multiclass SVM can train a single model that completely solves a multiclass classification problem. It can be seen an optimized implement of 1vsRest SVMs, with almost the same accuracy but much less time and memory cost.

The key parameter of a multiclass SVM is the trade off between training errors and margins, usually noted as C, which is the same as a common SVM. We search for optimal C on each node of the decision tree with the methods of training and validation, that is, we first extract the involved samples from the global training set and validation set, and then train and classify on these two extracted subsets. Such process is done automatically in our system, and we set a minimum value of 5000 for the sake of robust. The final classifying system is trained on a union set of the training set and the validation set, which has 1.5 times number of samples over the original training set. To train such a system, we correspondingly multiply the optimal C's with a factor of 1.2, which is decided empirically by several actual experiments of submission.

The linear kernel of the multiclass SVM is used by us. As for the other settings of the multiclass SVM, the default values of the package are used. We have tried some other options of learning algorithm provided by the multiclass SVM package, but no higher accuracy has been achieved.

2 Computational Complexity Estimates

Hierarchical classification reduces the computational complexity of a large classification problem, which can be seen as an effect of dividing it into many smaller ones. In hierarchical classification, training the root classifier is dealing with all the samples but fewer target classes, and training the rest nodes are dealing with both fewer samples and fewer classes.

The complexity of our method is similar to the case discussed in Yang's paper except the base classifiers [3]. Yang uses 1vsRest binary classifiers as the nodes of a decision tree, while we use multiclass SVMs instead. The complexity of 1vsRest binary SVMs for a multiclass classification is [3]

$$O\left(n^{\alpha}m\right), 1 < \alpha < 2 \tag{1}$$

where n is the number of samples, m is the number of classes, and α is a constant.

In the following passages, we will discuss the complexity of multiclass SVMs and the complexity of hierarchical classification.

2.1 Computational Complexity of Multiclass SVMs

The algorithm of training multiclass SVMs is an iterative one which terminates when a pre-defined error tolerance is satisfied, thus its actual memory and time cost is not very certain. Joachims proves its computation complexity to be linear to the number of samples (not strictly) and polynomial to some others factors, but unexpectedly not related to the number of classes [2]. However, the complexity formula given by him is a bit complex as it has several auxiliary variables, which is not suitable for complexity estimates.

Empirically, a multiclass SVM runs many times faster than a batch of binary SVM under 1vsRest framework in solving a multiclass classification problem. This may be because that a multiclass SVM handles all the classes at the same time so it can save much common computation.

Based on Joachims' proof, our experimental experience and (1), the computational complexity of a multiclass SVM is

$$O\left(n^{\alpha}m^{\beta}\right), 1 < \alpha < 2, 0 < \beta < 1 \tag{2}$$

where n is the number of samples, m is the number of classes, and α and β are two constants.

2.2 Computational Complexity of Hierarchical Classification

By applying (2), the complexity of training for hierarchical classification is (similar with [3])

$$\sum_{i=0}^{h-1}\sum_{j=0}^{l_i} O\left(n_{ij}^{\alpha} m_{ij}^{\beta}\right) = \sum_{i=0}^{h-1} O\left(N_i^{\alpha}\right) \sum_{j=0}^{l_i} O\left(\pi_{ij}^{\alpha} m_{ij}^{\beta}\right)$$
$$\leq O\left(N_0^{\alpha}\right) \sum_{i=0}^{h-1}\sum_{j=0}^{l_i} O\left(\pi_{ij}^{\alpha} m_{ij}^{\beta}\right) \quad (3)$$

where

- h is the depth of the hierarchy;
- l_i is the number of classes at the i^{th} level;
- $i = 0, 1, \ldots,$ h, and $i = 0$ corresponds to the root level;
- $j = 1, 2, \ldots, l_i$ are the classes at the i^{th} level
- n_{ij} is the number of local training samples;
- m_{ij} is the number of local classes;
- $N_i = \sum_{j=0}^{l_i} n_{ij}$ and we have $N_i \leq N_0$
- $\pi_{ij} = \frac{n_{ij}}{N_i}$ and $\sum_{j=1}^{l_i} \pi_{ij}$=1

The computational complexity of hierarchical classification depends on the hierarchies. Just to demonstrate how (3) actually reduces the complexity, suppose all the internal nodes have m_0 children, and all the nodes in the level has the same number of samples, then we can get

$$O\left(N_0^{\alpha}\right) m_0^{\beta} \left(1 + \frac{1}{m_0^{\alpha-1}} + \frac{1}{m_0^{2(\alpha-1)}} + \ldots + \frac{1}{m_0^{(h-1)(\alpha-1)}}\right) < O\left(h N_0^{\alpha} M_0^{\frac{\beta}{h}}\right) \quad (4)$$

where N_0 and M_0 is the total number of samples and classes, h is the depth of the tree. We can find out that the complexity is reduced in hierarchical classification by comparing (2) and (4).

3 Experiments

This section first presents our system's results on the dry-run test and on the formal test, and then describes the hardware that runs our system and the correspondent training time.

3.1 Results of Dry-run Tests

Table 1 shows our system's results on the dry-run datasets. These results are a little lower than those of the formal task, which may be caused by two reasons.

First, the training set which is actually used in the large dataset is relatively larger; because both the training set and the validation set are used for training, while only the training set is used for training in the dry-run dataset. Second, optimal parameters are searched much more carefully in the large dataset than they are in the dry-run dataset.

Table 1. Results of dry-run test

Task	Accuracy	Macro-F_1	Macro-P	Macro-R
basic	0.413	0.251	0.238	0.300
cheap	0.354	0.214	0.207	0.252

Dry-run dataset is a great help for us to develop the classifying system, on which we can quickly test our ideas. Such methods as indexing with TFIDF, simplifying hierarchies and automatically searching for parameters of C are all first proved to be effective in the dry-run dataset, then they are applied in the large dataset.

3.2 Results of Formal Tests

Table 2 shows our system's performances on the main dataset, the accuracies of which are at the third and second position of all the participants.

Table 2. Test accuracy on the main dataset

Task	Accuracy	Macro-F_1	Macro-P	Macro-R	Tree induced error
basic	0.443	0.320	0.303	0.338	3.293
cheap	0.366	0.250	0.237	0.264	3.967

3.3 Hardware and Training Time

Our system is run on a 64bit personal computer, which has an Intel Quad CPU of 2.83Ghz, and 8.0G physical memories. The system is coded with the computer languages of Python and C. The Joachims' multiclass SVM has been rewritten on the parts of reading samples and classes so that it can be used for hierarchical classification in an efficient manner. Table 3 shows the time cost of our system on the official submission tasks, that is, training on the large training and validation sets and classifying on the large test set. If a computer with enough physical memories is used, the training time of the basic task might be reduced to about 8 hours, which is close to that of the cheap task. It is because that the memory requirement of training SVMs for the basic task is close to 18G, well above our computer, so a lot of much extra time is cost by using virtual memories.

Table 3. time cost on the large datasets

Task	Training time(sequential)	Classifying time
basic	about 18 hours	12 minutes
cheap	about 6 hours	11 minutes

4 Conclusions and Future Work

We develop a decision tree of multiclass SVMs for large-scale hierarchical text classification, and improve our system mostly through simplifying the hierarchical structures. We have the following conclusions through this research.

1. Hierarchical classification is a practical solution to large-scale text classification, that is, a large number of samples and classes. It is difficult to directly apply SVMs to the problem without the hierarchical framework.
2. Hierarchical classification may reduce accuracy compared with flat classification, and simplifying the hierarchy can help to raise the accuracy.

In the future, we will explore better ways to incorporate multiclass SVM into hierarchical classification for less time cost and higher accuracy. For example, we can make the training and classifying between a parent node and its children nodes interactive. Optionally, we will try to convert multiclass SVM into a hierarchical algorithm; though Joachims et al provided a solution on this, the memory requirement makes it impractical for large-scale problems [2].

References

1. Sebastiani, F.: Machine learning in automated text categorization, ACM Computing Surveys, vol. 34, pp. 1-47, 2002.
2. I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun: Support Vector Machine Learning for Interdependent and Structured Output Spaces. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2004.
3. Yang, Y.M., Zhang, J., and Kisiel, B.: A Scalability Analysis of Classifiers in Text Categorization. In *ACM SIGIR'03*, pp 96–103, 2003.
4. Koller,D., and Sahami, M.: Hierarchically classifying documents using very few words. In *Proceedings of the 14th International Conference on Machine Learning*, pp 170–178, 1997.
5. DÁlessio,S., Murray,K., and Schiaffino,R.: The Effect of Using Hierarchical Classifiers in Text Categorization. In *Assistée par Ordinateur RIAO*,pp 302–313, 2000.
6. Ceci, M., and Malerba, D.: Classifying Web Documents in a Hierarchy of Categories: a Comprehensive Study. In *Journal of Intelligent Information Systems*,Vol.28, pp 37–38, 2007.