Ensembles of Binary SVM Decision Trees

Gjorgji Madjarov, Dejan Gjorgjevikj and Tomche Delev

Faculty of Electrical Engineering and Information Technologies, Ss. Cyril and Methodius University, Rudjer Boskovic bb, 1000 Skopje, Macedonia
{madzarovg, dejan, tdelev}@feit.ukim.edu.mk

Abstract. Ensemble methods are able to improve the predictive performance of many base classifiers. In this paper, we consider two ensemble learning techniques, bagging and random forests, and apply them to Binary SVM Decision Tree (SVM-BDT). Binary SVM Decision Tree is a tree based architecture that utilizes support vector machines for solving multiclass problems. It takes advantage of both the efficient computation of the decision tree architecture and the high classification accuracy of SVMs. In this paper we empirically investigate the performance of ensembles of SVM-BDTs. Our most important conclusions are: (1) ensembles of SVM-BDTs yield noticeable better predictive performance than the base classifier (SVM-BDT), and (2) the random forests ensemble technique is more suitable than bagging for SVM-BDT.

Keywords: Ensembles, Bagging, Random Forests, Support Vector Machines, Binary decision tree.

1 Introduction

The recent results in pattern recognition have shown that support vector machine (SVM) [1][2][3] classifiers often have superior recognition rates in comparison to other classification methods. However, the SVM was originally developed for binary decision problems, and its extension to multi-class problems is not straightforward. The popular methods for applying SVMs to multiclass classification problems usually decompose the multi-class problems into several two-class problems that can be addressed directly using several SVMs. Similar to these methods, we have developed an architecture of SVM classifiers utilizing binary decision tree (SVM-BDT) for solving multiclass problems [4]. This architecture uses hierarchy clustering algorithm to convert the multi-class problem into a binary tree of SVMs. The binary decisions in the non-leaf nodes of the binary tree are made by the SVMs. The SVM-BDT architecture [4] uses Euclidean distance in the clustering process for measuring the similarity between classes.

The goal of this paper is to investigate whether ensemble methods [5] can be applied to SVM-BDT in order to achieve better performance. Ensemble methods construct a set of classifiers for a given prediction task and classify new data instances by taking a vote over their predictions. Ensemble methods typically improve the predictive performance of their base classifier [5]. In this paper, we use SVM-BDT as base classifiers. The ensemble methods that we investigate are bagging [5] and random forests [6]. More precisely, the main question we want to answer is: Does building ensembles of SVM-BDTs really improve the predictive performance and which of the two ensemble techniques is more suitable for SVM-BDT. The last comparison is made along running times (training and testing).

The paper is organized as follows. In Section 2, we briefly discuss ensemble methods. Section 3 explains SVM-BDT in more detail. Section 4 presents a detailed experimental evaluation. Conclusions and some ideas for further work are presented in Section 5.

2 Ensemble methods

An ensemble is a set of classifiers constructed with a given algorithm. Each new example is classified by combining the predictions of every classifier from the ensemble. These predictions can be combined by taking the average (for regression tasks) or the majority vote (for classification tasks), as described by Breiman [5], or by taking more complex combinations [7][8][9]. A necessary condition for an ensemble to be more accurate than any of its individual members, is that the classifiers are accurate and diverse [10]. An accurate classifier does better than random guessing on new examples. Two classifiers are diverse if they make different errors on new examples. There are several ways to introduce diversity: by manipulating the training set (by changing the weight of the examples [5][11] or by changing the attribute values of the examples [12]), or by manipulating the learning algorithm itself [13]. In this paper, we consider two ensemble learning techniques that have primarily been used in the context of decision trees: bagging and random forests.

2.1 Bagging

Bagging [5] is an ensemble method that constructs the different classifiers by making bootstrap replicates of the training set and using each of these replicates to construct one classifier. Each bootstrap sample is obtained by randomly sampling training instances, with replacement, from the original training set, until an equal number of instances are obtained. Breiman [5] has shown that bagging can give substantial gains in predictive performance, when applied to an unstable learner (i.e., a learner for which small changes in the training set result in large changes in the predictions).

2.2 Random Forest

A random forest [6] is an ensemble of classifiers, where diversity among the predictors is obtained by using bagging, and additionally by changing the feature set during learning. More precisely, for each base classifier (SVM-BDT) in the ensemble, a random subset of the input attributes is taken. The number of attributes that are retained is given by a function *f* of the total number of input attributes *x* (e.g., $f(x) = 1$, $f(x) = x^{1/2}$, $f(x) = log_2(x) + 1 \ldots$). By setting $f(x) = x$, we obtain the bagging procedure.

3 Support Vector Machines Utilizing a Binary Decision Tree

SVM-BDT (Support Vector Machines utilizing Binary Decision Tree) [4] is tree based architecture which contains binary SVM in the non leaf nodes. It takes advantage of both the efficient computation of the tree architecture and the high classification accuracy of SVMs. Utilizing this architecture, *N*-1 SVMs are needed to be trained for an *N* class problem, but only log_2N SVMs in average are required to be consulted to classify a sample. This leads to a dramatic improvement in recognition speed when addressing problems with big number of classes.

The hierarchy of binary decision subtasks should be carefully designed before the training of each SVM classifier. There exist many ways to divide *N* classes into two groups, and it is critical to have proper grouping for the good performance of SVM-BDT. The SVM-BDT method is based on recursively dividing the classes in two disjoint groups in every node of the decision tree and training a SVM that will decide in which of the groups the incoming unknown sample should be assigned. The groups are determined by a clustering algorithm according to their class membership and their interclass distance. SVM-BDT method starts with dividing the classes in two disjoint groups g_1 and g_2. This is performed by calculating *N* gravity centers for the *N* different classes and the interclass distance matrix. Then, the two classes that have the biggest Euclidean distance from each other are assigned to each of the two clustering groups. After this, the class with the smallest distance from one of the clustering groups is found and assigned to the corresponding group. The gravity center of this group and the distance matrix are then recalculated to represent the addition of the samples of the new class to the group. The process continues by finding the next unassigned class that is closest to either of the clustering groups, assigning it to the corresponding group and updating the group's gravity center and distance matrix, until all classes are assigned to one of the two possible groups. This defines a grouping of all the classes in two disjoint groups of classes. This grouping is then used to train a SVM classifier in the root node of the decision tree, using the samples of the first group as positive examples and the samples of the second group as negative examples. The classes from the first clustering group are being assigned to the left sub-tree, while the classes of the second clustering group are being assigned to the right sub-tree.

The process continues recursively (dividing each of the groups into two subgroups applying the procedure explained above), until there is only one class per group which defines a leaf in the decision tree. The recognition of each sample starts at the root of the tree. At each node of the binary tree a decision is being made about the assignment of the input pattern into one of the two possible groups represented by transferring the pattern to the left or to the right sub-tree. This is repeated recursively downward the tree until the sample reaches a leaf node that represents the class it has been assigned to.

An example of SVM-BDT that solves a 7 - class pattern recognition problem utilizing a binary tree, in which each node makes binary decision using a SVM is shown on Fig. 1. a, while Fig. 1. b illustrates grouping of 7 classes.

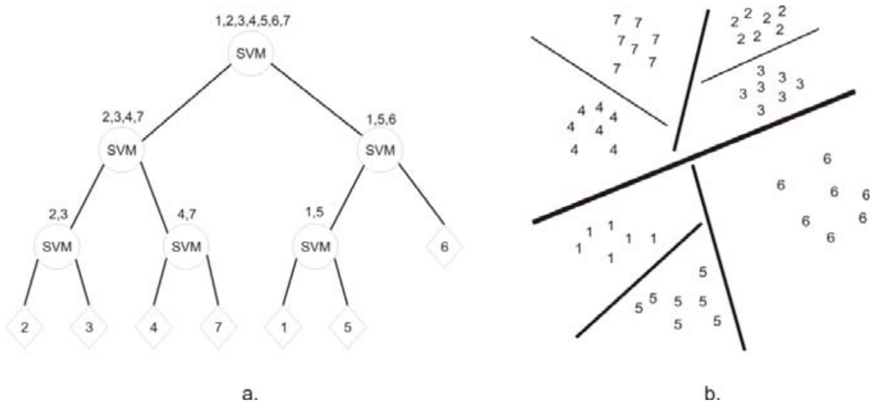

Fig. 1. a. SVM-BDT architecture; b. SVM-BDT divisions of seven classes

4 Experimental Evaluation

In this section, we describe the experimental methodology, the datasets, and the obtained results. The performances were measured on the problem of recognition of digits, letters and medical images.

We empirically evaluate two ensemble learning techniques, bagging and random forests, and apply them to Binary SVM Decision Tree (SVM-BDT). In order to combine the predictions output by the base classifiers, we apply the simple majority vote method. Each ensemble consists of 100 trees. For building random forests, the parameter $f(x)$ was set to $\lfloor log_2 x \rfloor$. The performances of the ensembles were also compared to the performance of the base classifier.

In our experiments, five different multi-class classification problems were addressed by each method (two ensemble methods and the base classifier SVM-BDT). The training and testing time and the recognition performance were recorded for every method.

The first problem was recognition of isolated handwritten digits from the MNIST database [14]. The MNIST database contains grayscale images of isolated handwritten digits. From each digit image, after performing a slant correction, 40 features were extracted. The features are consisted of 10 horizontal, 8 vertical and 22 diagonal projections [15]. The second and the third problem are 10 class problems from the UCI Repository [16] of machine learning databases: Optdigit and Pendigit.

The fourth problem was recognition of isolated handwritten letters, a 26-class problem from the Statlog collection [17]. The fifth problem was recognition of medical images, a 197-class problem from the IRMA2008 collection [18]. The medical images were described with 80 features obtained by the edge histogram descriptor from the MPEG7 standard [19]. The complete description of the datasets (number of classes, number of features, number of training and testing samples) is shown in Table 1.

Table 1. Datasets description

Dataset	# of classes	# of features	# of training samples	# of testing Samples
MNIST	10	40	60000	10000
Pendigit	10	16	7494	3498
Optdigit	10	64	3823	1797
Statlog	26	16	15000	5000
IRMA2008	197	80	12706	1000

In all classification problems the classifiers were trained using all available training samples of the sets and were evaluated by recognizing all the test samples from the corresponding set. All tests were performed on a personal computer with an Intel Core2Duo processor at 1.86GHz and 3GB of RAM under the Windows XP operating system.

The training and testing of the ensembles and base classifier were performed using a custom developed application that uses the Torch3 library [20]. The Torch3 library utilizing the SVMs with Gaussian kernel were used for solving the partial binary classification problems, we have used.

Table 2 through Table 4 shows the results of the experiments using the application of bagging and random forests and the base classifier (SVM-BDT) on each of the 5 data sets. Table 2 gives the prediction error rate of the ensembles and the base classifier applied on each of the datasets. Table 3 and Table 4 show the testing and training time of the ensembles and the base classifier, for the datasets, measured in seconds, respectively.

The results in Table 2 show that for all datasets, the ensemble methods achieved better prediction accuracy comparing to SVM-BDT base classifier. The error rates achieved by both ensemble methods are very similar. However, the bagging method achieved slightly lower error rates for all datasets except for the Optdigit dataset.

Table 2. The prediction error rate %

Dataset	SVM-BDT	Bagging	Random Forests
MNIST	2.45	1.88	1.90
Pendigit	1.94	1.89	1.92
Optdigit	1.61	1.26	1.24
Statlog	4.54	2.88	2.90
IRMA2008	55.80	48.00	48.00

Table 3. Testing time in seconds

Dataset	SVM-BDT	Bagging	Random Forests
MNIST	25.33	2312.76	419.67
Pendigit	0.54	48.54	32,17
Optdigit	0.70	72.46	31.12
Statlog	13.10	1145.82	603.12
IRMA2008	6.50	548.66	195.33

Table 4. Training time in seconds

Dataset	SVM-BDT	Bagging	Random Forests
MNIST	304.25	28455.96	16345.3
Pendigit	1.60	148.34	126,22
Optdigit	1.59	182.45	131.43
Statlog	63.30	6302.67	4252.13
IRMA2008	75.10	6934.59	2884.35

The testing and the training times are shown in Table 3 and Table 4. As expected, the training and testing times for the bagging method are about 100 times slower than the times of a single base classifier for all datasets (the number of the base classifiers in the ensembles was 100). The random forest ensemble is slightly faster than Bagging when training and considerably faster while testing. We believe this is to the smaller size of the feature vector used in the recognition process of the random forests method. The obtained results show that the dependency of the training and testing times of the SVMs in the binary decision trees are not proportional to the size of the reduced feature vector used by the random tree ensemble.

In overall, although the bagging ensemble method has shown slightly better recognition accuracy, the random forests is more suitable ensemble method for SVM-BDT because of its considerably lower time complexity.

5 Conclusion

In this paper, an empirical study on applying ensemble methods to SVM-BDT is presented. The results can be summarized as follows. First, the performance of a SVM-BDT is improved by learning an ensemble (using bagging or random forests) of SVM-BDTs. Second, the random forests ensemble technique is more suitable ensemble technique than bagging for SVM-BDT because of its considerably lower time complexity.

Although the time complexity of random forests ensemble method is lower, it is important to consider that the dependency of the training and testing times of the SVMs in the binary decision trees are not proportional to the size of the reduced feature vector used by this ensemble method.

As future work, we plan to extend the empirical evaluation along two dimensions: (a) how different feature selection strategies can influence on random forests ensemble method; and (b) how boosting ensemble methods can be applied to SVM-BDT.

References

1. Vapnik, V.: The Nature of Statistical Learning Theory, Springer, New York, (1999)
2. Burges, C., J., C.: A tutorial on support vector machine for pattern recognition. Data Min. Knowl. Disc. 2 (1998) 121
3. Joachims, T.: Making large scale SVM learning practical. in B. Scholkopf, C. Bruges and A. Smola (eds). Advances in kernel methods-support vector learning, MIT Press, Cambridge, MA, (1998)
4. Madzarov, G., Gjorgjevikj, D., Chorbev, I.: A multi-class SVM classifier utilizing binry decision tree, An International Journal of Computing and Informatics, Informatica, Volume 33 Number 2, ISSN 0350-5596, Slovenia, (2009) 233-241
5. Breiman, L.: Bagging predictors. Machine Learning 24(2) (1996) 123–140
6. Breiman, L.: Random forests. Machine Learning 45(1) (2001) 5–32
7. Ho, T., Hull, J., Srihari, S.: Decision combination in multiple classifier systems. IEEE Trans. on Pattern Anal. and Mach. Intell. 16(1) (1994) 66–75
8. Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. IEEE Trans. on Pattern Anal. and Mach. Intell. 20(3) (1998) 226–239
9. Gorgevik, D., Cakmakov D.: Combining SVM Classifiers for Handwritten Digit Recognition. Proceedings of 16th Int. Conference on Pattern Recognition, ICPR2002, Vol. 3, SII.8p, IEEE Computer Society, Quebec City, Canada, (2002)
10. Hansen, L., Salamon, P.: Neural network ensembles. IEEE Trans. on Pattern Anal. and Mach. Intell. 12 (1990) 993–1001
11. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proc. of the 13th ICML, Morgan Kaufmann (1996) 148–156
12. Breiman, L.: Using adaptive bagging to debias regressions. Technical report, Statistics Department, University of California, Berkeley (1999)
13. Dietterich, T.: Ensemble methods in machine learning. In: Proc. of the 1th Int'l Workshop on Multiple Classifier Systems. Volume 1857 of LNCS. (2000) 1–15
14. MNIST, MiniNIST, USA http://yann.lecun.com/exdb/mnist
15. Gorgevik, D., Cakmakov, D., An Efficient Three-Stage Classifier for Handwritten Digit Recognition, Proceedings of 17th ICPR2004. Vol. 4, IEEE Computer Society, Cambridge, UK, (2004) 507-510
16. Blake, C., Keogh, E., Merz, C. UCI Repository of Machine Learning Databases, (1998), http://archive.ics.uci.edu/ml/datasets.html [Online]
17. Statlog, http://archive.ics.uci.edu/ml/datasets/Letter+Recognition [Online]
18. http://www.imageclef.org/2008/medaat
19. Martinez, J. M. ed., MPEG Requirements Group, ISO/MPEG N4674, Overview of the MPEG-7 Standard, v 6.0, Jeju, Mar. 2002
20. Collobert, R., Bengio, S., Mariethoz., J.: Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP, 2002.