UNITECH '08 GABROVO

INTERNATIONAL SCIENTIFIC CONFERENCE
21 – 22 November 2008, GABROVO

MULTI-CLASS CLASSIFICATION USING SUPPORT VECTOR MACHINES IN BINARY TREE ARCHITECTURE

Gjorgji Madzarov, Dejan Gjorgjevikj, Ivan Chorbev
Department of Computer Science and Engineering
Faculty of Electrical Engineering and Information Technology
Ss. Cyril and Methodius University - Skopje, Macedonia
e-mail: {madzarovg, dejan, ivan}@feit.ukim.edu.mk

Abstract

This paper presents architecture of Support Vector Machine classifiers arranged in a binary tree structure for solving multi-class classification problems with increased efficiency. The proposed SVM based Binary Tree Architecture (SVM-BTA) takes advantage of both the efficient computation of the tree architecture and the high classification accuracy of SVMs. Clustering algorithm is used to convert the multi-class problem into binary tree, in which the binary decisions are made by the SVMs. The proposed clustering model utilizes distance measures at the kernel space, not at the input space. The performance of this method was measured on the problem of recognition of handwritten digits and letters using samples from MNIST, Pendigit, Optdigit and Statlog database of segmented digits and letters. The results of the experiments indicate that this method has much faster training and testing times than the widely used multi-class SVM methods like "one-against-one" and "one-against-all" while keeping comparable recognition rates. The experiments showed that this method becomes more favorable as the number of classes in the recognition problem increases.

Keywords: Support Vector Machine, multi-class classification, clustering, binary tree architecture.

INTRODUCTION

Recent results in pattern recognition have shown that SVM (Support Vector Machine) 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 straight-forward. How to effectively extend it to solve multi-class classification is still an on-going research issue. The popular methods for applying SVMs to multi-class classification problems usually decompose the multi-class problems into several two-class problems that can be addressed directly using several SVMs.

For the readers' convenience, we will introduce the SVM briefly in Section 2. A brief introduction to several widely used multi-class classification methods that utilize binary SVMs will be given in Section 3. The Kernel-based clustering introduced to convert the multi-class problems into SVM-based binary-tree architectures is explained in Section 4. The experimental results are presented to compare the performance of the proposed SVM-BTA with traditional multi-class approaches in Section 5. Section 6 gives a conclusion of the paper.

SUPPORT VECTOR MACHINES FOR PATTERN RECOGNITION

The support vector machine is originally a binary classification method developed by Vapnik and colleagues at Bell laboratories [1][2], with algorithm improvements by others [3]. SVM consists of projecting the input vectors into a high dimensional feature space, and then searching for the linear decision boundary that maximizes the minimum distance between two class groups (Figure 1).

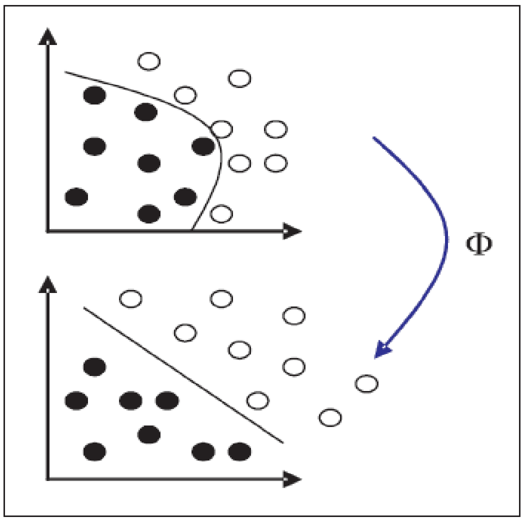

Fig. 1. General principle of SVM: projection

of data in an optimal dimensional space.

On Figure 1 we can see that data are not linearly separable in the initial space a) and after projection (by mapping Φ) they become separable in the high dimensional space b). SVM then consists of finding the optimal boundary for separating the positive class (dark circles) from the negative one (white circles).

SVM separates between these two classes via a hyperplane that is optimally positioned to maximize the margin between the positive samples and the negative ones (Figure 1), then 'plot' the test data at the high dimensional space, distinguishing whether it belongs to positive or negative side according to the optimal hyperplane.

For a binary classification problem with input space X and binary class labels Y:

$$Y \in \{-1, 1\} \tag{1}$$

Giving training samples (y_1, x_1), ..., (y_l, x_l).

$$y_i \in \{-1, 1\} \tag{2}$$

the goal of SVM is to search for the optimal hyperplane

$$w \cdot x + b = 0 \tag{3}$$

with variables *w* and *b* that satisfy the following inequality

$$y_i(w \cdot x_i + b) \geq 1, i = 1, \ldots, l, \tag{4}$$

defining the minimum distance between two class groups in the new projection.

$$d(w,b) = \min_{x/y=+1} \frac{x^T \cdot w}{\|w\|} - \max_{x/y=-1} \frac{x^T \cdot w}{\|w\|} \tag{5}$$

From e.q. (4), $\min_{\{x:y=1\}} x \cdot y = 1$ and $\min_{\{x:y=1\}} x \cdot y = -1$
Substituting back into e.q. (5), yields

$$d(w_0, b_0) = \frac{2}{\|w_0\|} = \frac{2}{\sqrt{w_0 w_0}}. \tag{6}$$

For a given training set *w*, *b* that maximizes $d(w_0, b_0)$ solve the following quadratic optimization problem:

$$\min_w \frac{1}{2} w \cdot w$$

satisfying $y_i(w \cdot x_i + b) \geq 1, i = 1, \ldots, l$ (7)

If the given training sample set is linearly separable, the optimization problem (7) has feasible solutions. The optimal solution *w* and *b* forms the best hyperplane that maximizes the margin between two different classes in the new projection. Because SVM search for the best separation hyperplane instead of the highest training sample accuracy, they never over-train on a sample data set. If the parameters are properly selected, SVM typically produces both excellent classification results and good generalization. Not every problem is guaranteed to be linearly separable, so a soft margin hyperplane SVM was developed to separate the training set with a minimal number of errors [4].
A number of candidate kernel functions have been used in SVM, including polynomial

$$K(x,y) = (1 + x \cdot y)^d \tag{8}$$

exponential RBF

$$K(x,y) = \exp\left(-\frac{\|x-y\|}{2\sigma^2}\right) \tag{9}$$

and Gaussian RBF

$$K(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) \tag{10}$$

For the new data point x, the classification is then performed as

$$y = sign(f(x)), \quad f(x) = \sum_{i=1}^{N_{SV}} \alpha_i y_i K(x, x_i) + b \tag{11}$$

Where N_{SV} is the number of support vectors.

OVERVIEW OF WIDELY USED MULTI-CLASS CLASSIFICATION METHODS

Although SVMs were originally designed as binary classifiers, approaches that address a multi-class problem as a single "all-together" optimization problem exist [5], but are computationally much more expensive than solving several binary problems.

A variety of techniques for decomposition of the multi-class problem into several binary problems using Support Vector Machines as binary classifiers have been proposed, and several widely used are:

- One-against-all

For the N-class problems (N>2), N 2-class SVM classifiers are constructed [4]. The *i*th SVM is trained while labeling all the samples in the *i*th class as positive examples and the rest as negative examples. In the recognition phase, a test example is presented to all N SVMs and is labeled according to the maximum output among the N classifiers. The disadvantage of this method is that the number of training samples is too large, so it is difficult to train.

- One-against-one

This algorithm constructs N(N-1)/2 2-class classifiers, using all the binary pair-wise combinations of the N classes. Each classifier is trained using the samples of the first class as positive examples and the samples of the second class as negative examples. To combine these classifiers, it naturally adopts Max Wins algorithm that finds the resultant class by first voting the classes according to the results of each classifier and then choosing the class that is voted most [6]. The disadvantage of this method is that every test sample has to be presented to large number of classifiers (N(N-1)/2). This results in faster training but slower testing, especially when the number of the classes in the problem is big [7].

- Directed acyclic graph SVM (DAGSVM)

Introduced by Platt [1] the algorithm for training a N(N-1)/2 classifiers is the same as in one-against-one. In the recognition phase, DAGSVM depends on a rooted binary directed acyclic graph to make a decision [8]. When a test sample reaches the leaf node, the final decision is made. A test example is presented only to the N-1 SVMs in the nodes on the decision path. This results in significantly faster testing while keeping very similar recognition rate as One-against-one.

- Binary Tree of SVM (BTS)

This method uses multiple SVMs arranged in a binary tree structure [9]. A SVM in each node of the tree is trained using two of the classes. The algorithm then employs probabilistic outputs to measure the similarity between the remaining samples and the two classes used for training. All samples in the node are assigned to the two subnodes derived from the previously selected classes by similarity. This step repeats on every node until each node contains only one class samples. The main problem that should be considered seriously here is training time, because, one has to test all samples in every node to find out which classes should be assigned to which subnode while building the tree. This may decrease the training performance considerably for huge training datasets.

In this paper we propose a binary tree architecture that uses SVMs for making the binary decisions in the nodes. The proposed classifier architecture SVM-BTA (Support Vector Machines with Binary Tree Architecture), 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 (like in one-aginst-all), but only $\lceil \log_2 N \rceil$ SVMs are required to be consulted to classify a sample. This can lead to a dramatic improvement in recognition speed when addressing problems with big number of classes.

SUPPORT VECTOR MACHINES IN BINARY TREE ARCHITECTURE (SVM-BTA)

As shown on Figure 2, the SVM-BTA solves an N-class pattern recognition problem utilizing a binary tree, in which each node makes binary decision using a SVM. The hierarchy of binary decision subtasks should be carefully designed before the training of each SVM classifier.

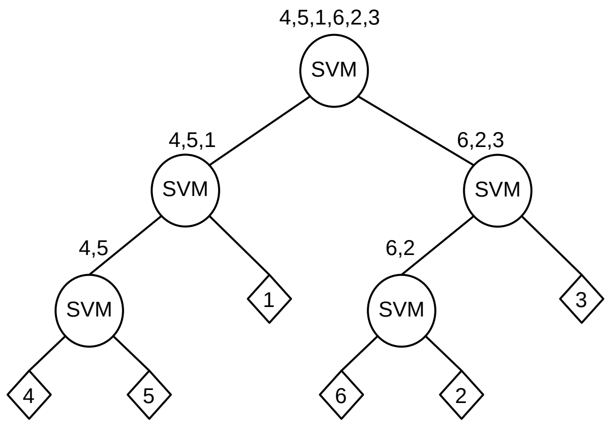

Fig 2. Illustration of SVM-BTA

The recognition of each pattern 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 right sub-tree. Each of these groups may contain multiple classes. This is repeated downward the tree until the sample reaches a leaf node that represents the class it has been assigned to.

There exist many ways to divide the classes into 2 groups, and it is critical to have proper grouping for the good performance of SVM-BTA.

For consistency between the clustering model and the way SVM calculates the decision hyperplane the clustering model utilizes distance measures at the kernel space, not at the input space. Because of this, all training samples are modified with the same kernel function that is to be used in the training phase.

The SVM-BTA method that we propose consists of two major steps: (1) computing a clustering of the classes and (2) associating a SVM at each node of the taxonomy obtained by (1).

Let's take a set of samples $x_1, x_2, ..., x_n$ labeled each one by $y_i \in \{c_1, c_2, ..., c_k\}$ where k is the number of classes. The first step of SVM-BTA method consists of calculating the gravity centers for the k known classes. Distance matrix with dimension $k \times k$ is created and its cells are filled with the Euclidean distances between the centers of i-th class and j-th class, where i and j are cell indexes. We start with randomly selecting one class from the classes in the corresponding node and adding it into a list. Then the matrix is searched for a class that has smallest distance to the already selected class. This class is added into the same list too. The algorithm proceeds with searching the nearest class to the last added class into the list which is not already added in it. This process continues until all the classes in the corresponding node are added to the list.

We always tend the binary tree for the SVM-BTA to be as balanced as possible, leading to best decision efficiency. To accomplish this, the classes from the first half of the list are assigned to the first group and the classes from the second half to the second group.

In the second step, each SVM is associated to a node and trained with the elements of the two groups of the corresponding node. For example, in Figure 2 which illustrates clustering of 6 classes, the SVM classifier in the root is trained by considering samples from the classes $\{c_1, c_4, c_5\}$ as positives examples and samples from the classes $\{c_2, c_3, c_6\}$ as negative examples. The SVM classifier in the left child of the root is then trained by considering samples from the classes $\{c_4, c_5\}$ as positives and samples from the class c_1 as negative examples. The concept is repeated for each SVM associated to a node in the taxonomy. This will result in training only $k-1$ SVMs for solving a k-class problem.

EXPERIMENTS

In this section, we present the results of our experiments with several multi-class problems. The performance was measured on the problem of recognition of handwritten digits and letters.

Training and testing of the SVMs was performed using a custom developed application that uses the Torch library [10]. For solving the partial binary classification problems SVMs using Gaussian kernel were used.

Here, we compare the results of the proposed SVM-BTA method with the following methods:

1) one-against-all (OvA);
2) one-against-one (OvO);
3) DAGSVM;
4) BTS;

The most important criterion in evaluating the performance of a classifier is usually its recognition rate, but very often the training and testing time of the classifier are equally important.

In our experiments 4 different multi-class classification problems were addressed by each of the 5 previously mentioned methods. For every method the training and testing time and the recognition performance were recorded.

The first problem was recognition of isolated handwritten digits from the MNIST database. The MNIST database [11] 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 [12]. The MNIST database contains 60.000 training samples, and 10.000 testing samples.

The second and the third problem are 10-class problems from the UCI Repository [13] of machine learning databases: optdigit and pendigit. Pendigit has 16 features, 7494 training samples, and 3498 testing samples. Optdigit has 64 features, 3823 training samples, and 1797 testing samples.

The fourth problem was recognition of isolated handwritten letters – a 26-class problem from the Statlog collection [14]. Statlog-letter contains 15.000 training samples, and 5.000 testing samples, while each sample is represented by 16 features.

The classifiers were trained using all available training samples for the set and were evaluated by recognizing all the test samples for the corresponding set. All tests were performed on personal computer with Intel Core2Duo processor at 1.86GHz on Windows XP.

Tables 1 to 4 show the results of the experiments using 5 different approaches on each of the 4 data sets. The first column of each of the tables describes the combining method of binary SVM classifiers: one-against-all (OvA), one-against-one (OvO), DAGSVM, BTS and SVM-BTA. In the second column the training parameters σ and C are given. The last three columns present the error-rate, the training time and the testing time for the corresponding method.

The results in table 1 show that for the MNIST database (10 classes, large number of samples) OvA method shows the lowest error rate, but is also slowest to train. The other methods show higher but similar error rates. The DAGSVM method shows fastest training and testing times.

Table 1. Recognition results, training and testing times for the MNIST dataset

Classifier	σ, C	Error-rate(%)	Time(s)	
			test	train
OvA	2, 100	1.93	23.56	468.94
OvO	2, 100	2.43	26.89	116.96
DAGSVM	2, 100	2.50	9.46	116.96
BTS	2, 100	2.24	26.89	240.73
SVM-BTA	2, 100	2.66	13.35	399.25

Table 2. Recognition results, training and testing times for the pendigit dataset

Classifier	σ, C	Error-rate(%)	Time(s)	
			test	train
OvA	60, 100	1.70	1.75	4.99
OvO	60, 100	1.94	3.63	3.11
DAGSVM	60, 100	1.97	0.55	3.11
BTS	60, 100	1.94	0.57	5.21
SVM-BTA	60, 100	1.91	0.61	1.62

Table 3. Recognition results, tarining and testing times for the optdigit dataset

Classifier	σ, C	Error-rate(%)	Time(s)	
			test	train
OvA	26, 100	1.17	1.68	3.92
OvO	26, 100	1.51	2.10	2.45
DAGSVM	26, 100	1.55	0.62	2.45
BTS	26, 100	1.51	0.65	4.68
SVM-BTA	26, 100	1.55	0.64	1.51

Table 4. Recognition results, tarining and testing times for the statlog dataset

Classifier	σ, C	Error-rate (%)	Time(s)	
			test	Train
OvA	1.1, 100	3.20	119.5	554.2
OvO	1.1, 100	4.72	160.5	80.9
DAGSVM	1.1, 100	4.74	12.5	80.9
BTS	1.1, 100	4.70	17.2	387.1
SVM-BTA	1.1, 100	4.48	13.2	63.7

From the results in table 2 and table 3 we can see that methods one-against-one (OvO), DAGSVM, BTS and our method SVM-BTA can reach almost the same accuracy. The method one-against-all (OvA) is more accurate than the other methods, which is apparent in both cases. Among all the methods, SVM-BTA is the fastest one in the training phase. Testing time is comparable in methods DAGSVM, BTS and SVM-BTA and they are noticeably better then testing time of one-against-all (OvA) and one-against-one (OvO) methods. However, if the number of the classes is relatively small, the advantage of SVM-BTA is not that evident.

For the three 10-class problems it can be noticed that OvA approach has the lowest error rate. On the other hand, the time needed to train the 10 classifiers for the OvA approach took about 4 times longer than training the 45 classifiers for the OvO and DAGSVM methods. The DAGSVM method showed to be the fastest in the recognition phase but also produces the biggest error rate.

The third problem was recognition of handwritten letters from the Statlog database [14]. Table 4 presents the results of the experiment for this 26-class problem. Again the OvA method showed the lowest error rate but the longest training time. The OvO, DAGSVM and the BTS method achieved very similar error rates that were about 1.5% higher than the OvA method. The DAGSVM is again fastest in recognition being almost 10 times faster than OvA. The time needed for training of the 26 one-against-all SVMs was almost 7 times longer than the time for training the 325 one-against-one SVMs. The BTS method showed the lowest error rate of the methods that use one-against-one SVMs. The SVM-BTA method showed better recognition rate than the methods using one-against-one SVMs while being only slightly slower in recognition than DAGSVM and the fastest while training.

CONCLUSION

We have presented a novel method of arranging a binary classifiers like support vector machines in order to solve a multi-class problem. The proposed Support Vector Machines in Binary Tree Architecture (SVM-BTA) method was designed to provide superior recognition speed utilizing decision tree architecture, while keeping comparable recognition rate to the other known methods. Clustering algorithm that utilizes distance measures at the kernel space is used to convert the multi-class problem into binary tree, in which the binary decisions are made by the SVMs. The experiments performed on 4 different datasets of handwritten digits and letters have shown that this method has one of the fastest training times while keeping similar recognition rate to the other methods. SVM-BTA is becoming more favorable to the other compared methods as the number of classes in the problem increases.

REFERENCE

[1] V. Vapnik, *The Nature of Statistical Learning Theory*, second ed., Springer, New York, 1999.

[2] C. J. C. Burges, *A tutorial on support vector machine for pattern recognition*, Data Min. Knowl. Disc. 2 (1998) 121.

[3] T. Joachims, 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] V. Vapnik. *Statistical Learning Theory,* Wiley, New York, 1998.

[5] J. Weston, C. Watkins, Multi-class support vector machines, *Proceedings of ESANN99*, M. Verleysen, Ed., Brussels, Belgium, 1999.

[6] J. H. Friedman, Another approach to polychotomous classification, Technical report, Department of Statistics, Stanford University, 1997.

[7] P. Xu, A. K. Chan, Support vector machine for multi-class signal classification with unbalanced samples, *Proceedings of the International Joint Conference on Neural Networks 2003*, Portland, pp.1116-1119, 2003.

[8] Platt, N. Cristianini and J. Shawe-Taylor, Large margin DAGSVM's for multiclass classification, *Advances in Neural Information Processing System*, Vol. 12, pp. 547–553, 2000.

[9] B. Fei, J. Liu, Binary Tree of SVM: A New Fast Multiclass Training and Classification Algorithm, *IEEE Transaction on neural networks*, Vol. 17, No. 3, May 2006.

[10] R. Collobert, S. Bengio and J. Mariéthoz, Torch: a modular machine learning software library, Technical Report IDIAP-RR 02-46, IDIAP, 2002.

[11] ____, MNIST, MiniNIST, USA http://yann.lecun.com/exdb/mnist

[12] D. Gorgevik, D. Cakmakov, An Efficient Three-Stage Classifier for Handwritten Digit Recognition, *Proceedings of 17th Int. Conference on Pattern Recognition, ICPR2004*, Vol. 4, pp. 507-510, IEEE Computer Society, Cambridge, UK, 23-26 August 2004.

[13] C. Blake, E. Keogh and C. Merz, UCI Repository of Machine Learning Databases, (1998). Statlog Data Set, http://www.ics.uci.edu/mlearn/MLRepository.html [Online]

[14] Statlog Data Set, ftp://ftp.ncc.up.pt/pub/statlog/ [Online]