Evaluation of Distance Measures for Multi-class Classification in Binary SVM Decision Tree

Conference Paper · June 2010

DOI: 10.1007/978-3-642-13208-7_55 · Source: DBLP

Dejan Gjorgjevikj
Ss. Cyril and Methodius University in Skopje

Evaluation of Distance Measures for Multi-class Classification in Binary SVM Decision Tree

Gjorgji Madzarov and Dejan Gjorgjevikj

Department of Computer Science and Engineering, Ss. Cyril and Methodius
University, Karpos 2 bb Skopje, Macedonia
{madzarovg,dejan}@feit.ukim.edu.mk
www.feit.ukim.edu.mk

Abstract. Multi-class classification can often be constructed as a generalization of binary classification. The approach that we use for solving this kind of classification problem is SVM based Binary Decision Tree architecture (SVM-BDT). It takes advantage of both the efficient computation of the decision tree architecture and the high classification accuracy of SVMs. The hierarchy of binary decision subtasks using SVMs is designed with a clustering algorithm. In this work, we are investigating how different distance measures for the clustering influence the predictive performance of the SVM-BDT. The distance measures that we consider include Euclidian distance, Standardized Euclidean distance and Mahalanobis distance. We use five different datasets to evaluate the performance of the SVM based Binary Decision Tree architecture with different distances. Also, the performance of this architecture is compared with four other SVM based approaches, ensembles of decision trees and neural network. The results from the experiments suggest that the performance of the architecture significantly varies depending of applied distance measure in the clustering process.

Key words: Support Vector Machines, Binary tree architecture, Euclidian distance, Standardized Euclidean distance and Mahalanobis distance

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 binary tree. 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 classes similarity. Here, we consider two additional distance measures (Standardized Euclidean distance and Mahalanobis distance).

The remainder of this paper is organized as follows: Section 2 describes the SVM-BDT algorithm and the proposed distance measures. The experimental results in section 3 are presented to compare the performance of the SVM-BDT architecture with different distance measures and with traditional multi-class approaches based on SVM, ensemble of decision trees and neural network. Finally, conclusions are presented in Section 4.

2 Metodology

2.1 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 $log_2 N$ SVMs in average are required to be consulted to classify a sample. This lead 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 in kernel space.

SVM-BDT method starts with dividing the classes in two disjoint groups g_1 and g_2. This is performed by calculating N gravity centres for the N different classes and the interclass distance matrix. Then, the two classes that have the biggest (in the first case Euclidean, in the second case Standardized Euclidean and in the third case Mahalanobis) 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 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 first (left) sub-tree, while the classes of the second clustering group are being assigned to the (right) second 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.

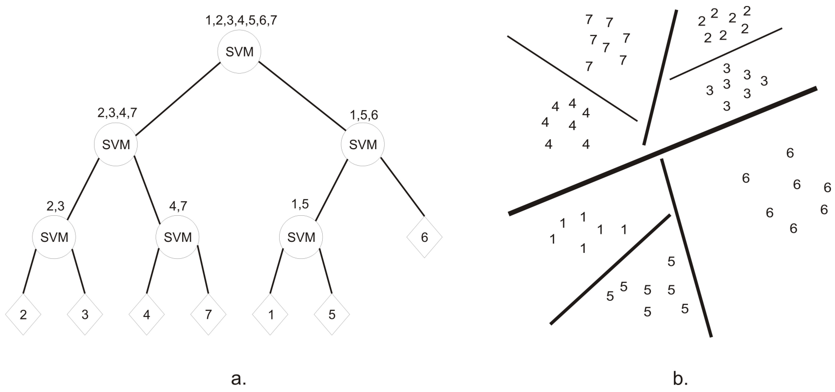

Fig. 1. a. SVM-BDT architecture; b. SVM-BDT divisions of seven classes

2.2 Euclidean Distance

Euclidean Distance is the most common used distance measure. In most cases when people said about distance, they will refer to Euclidean distance. Euclidean distance or simply "distance" examines the root of square differences between coordinates of a pair of objects.

$$d_{ij} = \left(\Sigma\left(\hat{x_i} - \hat{x_j}\right)^2\right)^{\frac{1}{2}}. \qquad (1)$$

The gravity centers of the two groups that are obtained by the clustering algorithm in the non leaf nodes of the tree are represented by $\hat{x_i}$ and $\hat{x_j}$.

2.3 Standardized Euclidean Distance

The contribution of each feature is different if the distance between two groups is measured by Euclidean Distance. Some form of standardization is necessary to balance out these contributions. The conventional way to do this is to transform the features so they all have the same variance of one. Euclidean Distance calculated on standardized data is called Standardized Euclidean Distance. This distance measure between two groups of samples can be written as:

$$d_{ij} = \left(\Sigma \left(\frac{\hat{x_i}}{\hat{s_i}} - \frac{\hat{x_j}}{\hat{s_j}} \right)^2 \right)^{\frac{1}{2}}, \tag{2}$$

where $\hat{s_i}$ and $\hat{s_j}$ are the group i and the group j standard deviation vectors respectively. The $\hat{x_i}$ and $\hat{x_j}$ are the gravity centers of the group i and group j, that are obtained by the clustering algorithm in the non leaf nodes of the tree.

2.4 Mahalanobis Distance

Mahalanobis distance [5] is also called quadratic distance. It measures the separation of two groups of samples. It differs from Euclidean distance in that it takes into account the correlations of the data set and is scale-invariant. Suppose we have two groups with means $\hat{x_i}$ and $\hat{x_j}$, Mahalanobis distance is given by

$$d_{ij} = \left((\hat{x_i} - \hat{x_j})^T S^{-1} (\hat{x_i} - \hat{x_j}) \right)^{\frac{1}{2}}, \tag{3}$$

where S^{-1} is an inverse pooled covariance matrix. This matrix is computed using weighted average of covariance matrices of both of the groups.

3 Experimental Results

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 digits, letters and medical images.

Here, we compare the results obtained by the SVM-BDT method with three different distance measures (Euclidean Distance - SVM-BDT$_E$, Standardized Euclidean Distance - SVM-BDT$_{SE}$ and Mahalanobis Distance - SVM-BDT$_M$) that are used in the clustering process. Also, the performance of this architecture is compared with the one-against-all (OvA) [6], one-against-one (OvO) [7][8], DAGSVM [9], BTS [10], Bagging [11], Random Forests [11], Multilayer Perceptron (MLP, neural network).

The training and testing of the SVMs based methods (SVM-BDT$_E$, SVM-BDT$_{SE}$, SVM-BDT$_M$, OvO, OvA, DAGSVM and BTS) was performed using a custom developed application that uses the Torch library [13]. For solving the partial binary classification problems, we used SVMs with Gaussian kernel. In these methods, we had to optimize the values of the kernel parameter σ and penalty C. For parameter optimization we used experimental results.

We also developed an application that uses the same (Torch) library for the neural network classification. One hidden layer with 25 units was used by the neural network. The number of hidden units was determined experimentally.

The classification based on ensembles of decision trees [11] (Bagging and Random Forest) was performed by Clus, a popular decision tree learner based on the principles stated by Blockeel et al. [12]. There were 100 models in the ensembles. The pruning method that we used was C4.5. The number of selected features in the Random Forest method was log_2K where K is the number of features in the dataset.

In our experiments, five different multi-class classification problems were addressed by each classifying methods. The training and testing time and the recognition performance were recorded for every method.

The first problem was recognition of isolated handwritten digits (10 classes) 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 (64 features) and Pendigit (16 features). The fourth problem was recognition of isolated handwritten letters, a 26-class problem from the Statlog (16 features) 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].

Table 1 through Table 3 show the results of the experiments using 10 different approaches (7 approaches based on SVM, two based on ensembles of decision trees and one neural network) on each of the 5 data sets. Primary we focused on the results achieved from SVM-BDT methods with Euclidean, Standardized Euclidean and Mahalanobis distance. Table 1 gives the prediction error rate of each method applied on each of the datasets. Table 2 and Table 3 shows the testing and training time of each algorithm, for the datasets, measured in seconds, respectively.

The results in the tables show that SVM based methods outperform the other approaches, in terms of classification accuracy. In terms of speed, SVM based methods are faster, with different ratios for different datasets. Overall, the SVM based algorithms were significantly better compared to the non SVM based methods.

The results in Table 1 show that for the MNIST, Pendigit and Optdigit datasets, the SVM-BDT$_M$ method achieved the best prediction accuracy comparing to SVM-BDT$_E$ and SVM-BDT$_{SE}$ methods. The results in Table 1, also show that for all datasets, the OvA method achieved the lowest error rate, except in the case of Pendigit dataset. It can be noticed that for the 197-class classification problem the prediction error rates, testing and training times of

Table 1. The prediction error rate % of each method for every dataset

	10-class			26-class	197-class
	MNIST	Pendigit	Optdigit	Statlog	IRMA2008
SVM-BDT$_E$	2.45	1.94	1.61	4.54	55.80
SVM-BDT$_{SE}$	2.43	1.90	1.65	4.55	55.00
SVM-BDT$_M$	2,15	1,63	1,55	4.54	/
OvO	2.43	1.94	1.55	4.72	/
OvA	1.93	1.70	1.17	3.20	48.50
DAGSVM	2.50	1.97	1.67	4.74	/
BTS	2.24	1.94	1.51	4.70	/
R. Forest	3.92	3.72	3.18	4.98	60.80
Bagging	4.96	5.38	7.17	8.04	64.00
MLP	4.25	3.83	3.84	14.14	64.00

Table 2. Testing time of each method for every dataset measured in seconds

	10-class			26-class	197-class
	MNIST	Pendigit	Optdigit	Statlog	IRMA2008
SVM-BDT$_E$	25.33	0.54	0.70	13.10	6.50
SVM-BDT$_{SE}$	24.62	0.55	0.71	13.08	6.45
SVM-BDT$_M$	20.12	0.61	0.67	12.90	/
OvO	26.89	3.63	1.96	160.50	/
OvA	23.56	1.75	1.63	119.50	19.21
DAGSVM	9.46	0.55	0.68	12.50	/
BTS	26.89	0.57	0.73	17.20	/
R. Forest	39.51	3.61	2.76	11.07	34.45
Bagging	34.52	2.13	1.70	9.76	28.67
MLP	2.12	0.49	0.41	1.10	0.60

Table 3. Training time of each method for every dataset measured in seconds

	10-class			26-class	197-class
	MNIST	Pendigit	Optdigit	Statlog	IRMA2008
SVM-BDT$_E$	304.25	1.60	1.59	63.30	75.10
SVM-BDT$_{SE}$	285.14	1.65	1.63	64.56	73.02
SVM-BDT$_M$	220.86	1.80	5.62	62.76	/
OvO	116.96	3.11	2.02	80.90	/
OvA	468.94	4.99	3.94	554.20	268.34
DAGSVM	116.96	3.11	2.02	80.90	/
BTS	240.73	5.21	5.65	387.10	/
R. Forest	542.78	17.08	22.21	50.70	92.79
Bagging	3525.31	30.87	49.4	112.75	850.23
MLP	45.34	2.20	1.60	10.80	42.43

the SVM-BDT$_M$, OvO, DAGSVM and BTS are left. These methods are uncompetitive to the other methods for this classification problem because of their long training time. In the first case the SVM-BDT$_M$ method took several hundred times longer training time. This appeared as a result of the calculation of the inverse pooled covariance matrix in the clustering process, because of the huge number of classes and the big number of features (80), which are characteristic for this classification problem. In the second case the one-against-one methods (OvO, DAGSVM and BTS) took long training and testing time, because of the large number of classifiers that had to be trained (19306) and the large number of classifiers that had to be consulted in the process of classification.

Of the non SVM based methods, the Random Forest method achieved the best recognition accuracy for all datasets. The prediction performance of the MLP method was comparable to the Random Forest method for the 10-class problems and the 197-class problem, but noticeably worse for the 26-class problem. The MLP method is the fastest one in terms of training and testing time, which is evident in Table 2 and Table 3.

The results in Table 2 show that the DAGSVM method achieved the fastest testing time of all the SVM based methods for the MNIST dataset. For the other datasets, the testing time of DAGSVM is comparable with BTS and SVM-BDT methods and their testing time is noticeably better than the OvA and OvO methods.

In terms of training speeds, it is evident in Table 3 that among the SVM based methods, SVM-BDT$_E$ is the fastest one in the training phase except for the MNIST dataset. Due to the huge number of training samples in the MNIST dataset (60000), SVM-BDT$_E$'s training time was longer compared to other one-against-one SVM methods. The huge number of training samples increases the nonlinearity of the hyperplane in the SVM, resulting in an increased number of support vectors and increased training time. Also, it is evident that the SVM-BDT$_M$ method is slower than the SVM-BDT$_E$ and the SVM-BDT$_{SE}$ methods in the training phase for the Optdigit classification problems. This appears as a result of the size of the feature vector (64) which is longer than the feature vectors of the other classification problems.

4 Conclusion

In this work, we have reviewed and evaluated several distance measures that can be applied in the clustering process of building the SVM-BDT architecture. In particular, we compared the Euclidean Distance, Standardized Euclidean Distance and Mahalanobis Distance. The predictive accuracy as a criterion of the performance of the classifiers shows that Mahalanobis Distance is the most suitable distance measure for measuring the similarity between classes in the clustering process of constructing the classifier architecture comparing to the other distance measures of the SVM-BDT methods. But, its training time complexity rapidly grows with the number of features of the classification problem and makes it uncompetitive to the other distance measure techniques like Euclidean and Standardized Euclidean Distances. The SVM-BDT$_E$ and the SVM-BDT$_{SE}$ show similar results for the predictive accuracy and also similar speed in the training and testing phase. Their complexities linearly depend from the characteristics of the classification problems. Comparing to the other SVM and non SVM based methods the SVM-BDT methods with different distance measure show comparable results or offer better recognition rates than the other multi-class methods. The speed of training and testing is improved when we used Euclidean Distance and Standardized Euclidean Distance for measuring the similarity between classes in the clustering process of constructing the classifier architecture.

References

1. V. Vapnik, The Nature of Statistical Learning Theory, 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. G. Madzarov, D. Gjorgjevikj, I. Chorbev, A multi-class SVM classifier utilizing binry decision tree, An International Journal of Computing and Informatics, Informatica, Volume 33 Number 2, ISSN 0350-5596, pp.233-241, Slovenia, 2009
5. P. Mahalanobis, On tests and measures of group divergence I. Theoretical formulae, J. and Proc. Asiat. Soc. of Bengal , 26 (1930) pp. 541588
6. V. Vapnik, Statistical Learning Theory. Wiley, New York, 1998
7. J. H. Friedman, Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University, 1997
8. P. Xu, A. K. Chan, Support vector machine for multi-class signal classification with unbalanced samples, Proceedings of the IJCNN2003. Portland, pp.1116-1119, 2003
9. Platt, N. Cristianini, J. Shawe-Taylor, Large margin DAGSVMs for multiclass classification, Advances in Neural Information Processing Sys. Vol. 12, pp. 547553, 2000
10. B. Fei, J. Liu, Binary Tree of SVM: A New Fast Multiclass Training and Classification Algorithm, IEEE Transaction on neural net., Vol. 17, No. 3, May 2006
11. D. Kocev, C. Vens, J. Struyf and S. Dzeroski, Ensembles of multi-objective decision trees, Proceedings of the 18th ECML (pp. 624631) (2007). Springer
12. H. Blockeel, J. Struyf, Efficient Algorithms for Decision Tree Cross-validation, Journal of Machine Learning Research 3:621-650, 2002
13. R. Collobert, S. Bengio, J. Mariethoz, Torch: a modular machine learning software library, Technical Report IDIAP-RR 02-46, IDIAP, 2002
14. MNIST, MiniNIST, USA http://yann.lecun.com/exdb/mnist
15. D. Gorgevik, D. Cakmakov, An Efficient Three-Stage Classifier for Handwritten Digit Recognition, Proceedings of 17th ICPR2004. Vol. 4, pp. 507-510, IEEE Computer Society, Cambridge, UK, 23-26 August 2004
16. C. Blake, E. Keogh and C. Merz, 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. J. M. Martinez, ed., MPEG Requirements Group, ISO/MPEG N4674, Overview of the MPEG-7 Standard, v 6.0, Jeju, Mar. 2002