ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning. Bruges (Belgium), 22-24 April 2009, d-side publi., ISBN 2-930307-09-9.

Embedding Proximal Support Vectors into Randomized Trees

Cedric Simon[1] and Jerome Meessen[2] and Christophe De Vleeschouwer[1]

1- UCL, Communication and Remote Sensing Lab
2, Place du Levant, 1348 Louvain-la-Neuve, Belgium

2- Multitel, Mons, Belgium

Abstract. By embedding multiple proximal SVM classifiers into a binary tree architecture, it is possible to turn an arbitrary multi-classes problem into a hierarchy of binary classifications. The critical issue then consists in determining in each node of the tree how to aggregate the multiple classes into a pair of say *overlay classes* to discriminate. As a fundamental contribution, our paper proposes to deploy an ensemble of randomized trees, instead of a single optimized decision tree, to bypass the question of overlay classes definition. Empirical results on various datasets demonstrate a significant gain in accuracy both compared to 'one versus one' SVM solutions and to conventional ensemble of decision trees classifiers.

1 Introduction

Support vector machine (SVM) has become a state of the art algorithm in data mining. It has been successfully applied to classification, feature selection, clustering or time series analysis. In a classification context, SVM inherently addresses binary class problems. Many methods have then been proposed to extend its use on multi-category datasets [3, 7], but very few among them proved to be more efficient than the "One Versus One" class methodology.

Recently, it has been suggested that SVM can benefit from the decision tree architecture to perform a more natural multi-category classification. The key idea consists in partitioning a set of multiple input classes into a pair of say *overlay classes*, which are then characterized based on a SVM binary classifier. In [3], Cheong has proposed a SVM-based binary tree architecture to take the best out of the computation performance of the tree architecture and of the high classification accuracy of SVMs. However, his main conclusions highlight the fact that converting the multi-class problem into a proper and optimal binary tree raises a very challenging and open question. Indeed, the definitions of overlay classes that are implemented in first nodes of the tree obviously affect subsequent classes aggregation options. This makes the derivation of an optimal decision tree structure quite complex, since it requires an exhaustive investigation of all possible classes aggregation options at each node of the tree. To alleviate this problem, Cheng and al. recently proposed a bottom up approach to determine the hierarchical partition of the multi-class problem into a sequence of binary SVMs [2]. The approach assumes that the first classes to isolate in the binary structure are the ones that have the largest average relative distance to other classes. Hence they follow a bottom up approach in which they first investigate how to separate most similar classes, as measured by some heuristic metric. Whilst possibly valuable, it is worth mentioning that such kind of heuristic process to guide the design of some optimal tree does not alleviate another well-known weaknesses of standalone decision trees. Specifically, optimized decision trees are known to offer poor generalization properties, i.e. they end up in over fitting the training data.

To circumvent those drawbacks, we propose to build an ensemble of randomized trees, instead of a single optimized decision tree. Our proposed approach has the advantage (1) to relax and virtually bypass the question associated to the definition of overlay classes in each individual tree, and (2) to increase the robustness and generalization capabilities of the resulting ensemble of classifiers.

Formally, each individual tree considers a pseudo-random partition of classes into two overlay classes in each one of its nodes. By randomizing the tree construction process and merging an ensemble of diversified trees to infer a class label, no strict decision has to be taken regarding the sequences of classes partitions encountered along the binary structure. The definition of overlay classes in each tree becomes part of the randomization process, and further helps in reducing the variance of the results [5]. The computational penalty induced by the use of multiple trees can be overcome by using proximal SVM instead of traditional SVM. Proximal SVM, which can also be interpreted as the regularized least squares solution of a linear equation system, was introduced by Suykens et al. [8] in order to speed up the computation of support vectors.

From the decision tree algorithm point of view, the use of support vectors at each node is an alternative at finding optimal thresholds to multiple attributes at each node. Recent works [6] showed that multiple attributes decision trees can improve the accuracy of the decision trees. Our results in the experimental section confirm and strengthen this assertion.

The rest of this paper is organized as follow. Section 2 presents our supervised classification algorithm, and related background theory. In section 3, experimental results are given on five machine learning datasets coming from the UCI repository. Finally, Section 4 concludes and gives some perspectives about future work.

2 Ensemble of proximal SVM trees

We propose to exploit the computational efficiency and the classification correctness of the proximal SVM to define discriminant and multiple attributes tests in the nodes of an ensemble of randomized decision trees. Randomization is known to generalization capabilities of the classifier. More importantly, in our framework, it permits to circumvent the question associated to the partition of the multiple categories at hand into a pair of overlay classes. Figure 1 shows the general architecture of the proposed classifier.

Linear PSVM We now rapidly review the keys lessons drawn from [4]. We consider the problem of classifying m points in the n-dimensional real space $\Re^n$, represented by the $m \times n$ matrix A, according to membership of each point A_i in

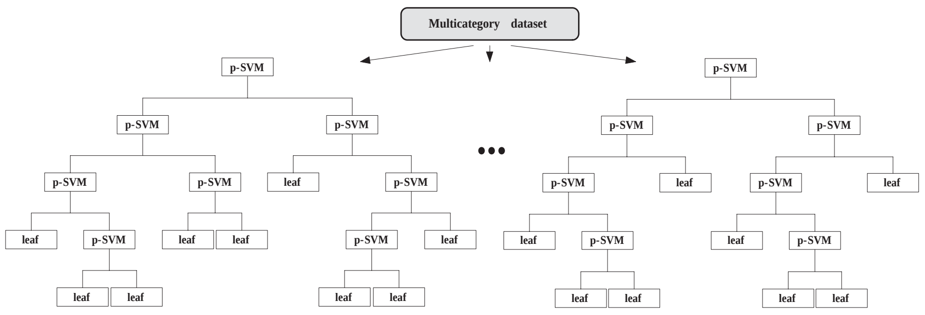

Fig. 1: Embedded proximal SVMs in an ensemble of decision trees

the positive or negative class, as specified by a given $m \times m$ diagonal matrix D with plus ones or minus ones along its diagonal. For this problem, the standard support vector machine with a linear kernel is given by the following quadratic program with parameter $\nu > 0$[9]:

$$\begin{aligned} \min_{(w,\gamma,y)\in\Re^{n+1+m}} & \nu e'y + \frac{1}{2}w'w \\ \text{s.t. } & D(Aw - e\gamma) + y \geq e \\ & y \geq 0 \end{aligned} \tag{1}$$

Geometrically, this formulation ends up in computing two planes $x'w - \gamma = \pm 1$ that are bounding most of the positive and negative data samples, respectively.

By minimizing the 2-norm of the error vector y (instead of the 1-norm), and by replacing the inequality constraint by an equality, Fung and Mangasarian [4] turn the SVM into the following *unconstrained* optimization problem:

$$\min_{(w,\gamma)\in\Re^{n+1+m}} \frac{\nu}{2} \parallel D(Aw - e\gamma) - e \parallel^2 + \frac{1}{2}(w'w + \gamma^2). \tag{2}$$

This new formulation can be seen as a regularized least squares solution of the system of linear equations $D(Aw - e\gamma) = e$. Hence, interestingly, from a computational point of view, this formulation replaces the resolution of a linear or quadratic program by the resolution of a nonsingular system of linear equations [4]. Geometrically, in this alternative formulation, the planes $x'w - \gamma = \pm 1$ can be thought of as 'proximal' planes, around which the points of each class are clustered and which are pushed as far apart as possible.

The resulting and so-called proximal SVM (PSVM) classifier has comparable test set correctness to that of standard SVM classifiers, but with considerably faster computational time that can be an order of magnitude faster. This computational efficiency is mandatory in our tree-based algorithm, where in some cases a few thousand SVM instances need to be created.

Embedding proximal SVM in a binary tree architecture Our system uses the classical top-down procedure to build unpruned decision trees [6] from the learning samples. Each node of a standard decision tree can be seen as a weak classifier that organizes the input samples into two branches so as to reduce the impurity of the output variable within the local learning subset. In our case, this partition is done based on a PSVM classifier. Once the entire tree has been grown, a sample to classify simply falls from the root to one of the leaves, and receives the label of the dominating class among the training samples that reached that leaf during the learning phase.

In order to use the proximal SVMs at each node of the trees, the multiple class labels of the training samples are converted into binary values, mapping the input classes to a pair of overlay classes. The only constraint imposed to the partition of the initial set in binary overlay classes intends to avoid strongly unbalanced dataset, so as to avoid significant reduction of the performance of the proximal SVM due to lack of samples in one of the overlay classes. Formally, input classes are progressively and randomly selected to feed the first overlay class until half of the data samples have been assigned the first overlay class label. The second label is then assigned to all other classes.

Ensemble of trees and random process Similar to numerous previous works [1, 5], randomization methods are considered to improve the tree decision accuracy. These methods explicitly randomize the tree growing algorithm, and perform multiple runs of the growing process, so as to produce an ensemble of more or less diversified tree models, whose predictions are aggregated by a simple majority vote. The purpose of multiple trees is to solve the approximation error and the estimation error problem at the same time.

In our framework, randomization has the important additional advantage of relaxing the impact of a specific decision about overlay classes partition (in a particular node of a particular tree) on the aggregated decision taken by the ensemble of classifiers. In that sense, it permits to decouple the tree building process from the overlay classes selection process. In practice, during our learning process, N_a attributes over n are selected at random to split the two pseudo-random overlay classes. Our trees nodes training procedure is detailed in Algorithm 1.

3 Empirical Results

In order to analyze the performance of the ensemble of linear Proximal SVM trees (PSVM Trees), we used various types of data sets to classify coming from the UCI Machine Learning Repository [1]. Those data sets propose 4 to 617 attributes, and 2 to 26 classes, thus covering a wide range of conditions.

For those experiments, we compare on table 3 our method to the ensemble of decision trees and SVM based methods, from which our method is based on. The linear multi-category proximal SVM (linear MPSVM) accuracies are taken from Fung and Mangasarian's paper [4], while the Extremely randomized Trees (i.e. the Extra-trees ET^d) performances were taken from Geurts paper [5]. We

[1]http://archive.ics.uci.edu/ml/

Algorithm 1 Training a node N

```
Split_a_node(S)
Input: a learning set of $m$ samples $S = (s_1, ..., s_m)$ with $n$ attributes, $N_c$ classes
Output: a node labeled by the PSVM $w$ and $\gamma$, the output branch of each learning sample
 1: $\{S_1, S_2\}$ = Split_in_2_categories(S)
 2: $m_1$ & $m_2$: quantities of learning samples in each category;
 3: $[a_1, ..., a_{N_a}]$ = Select_attributes($n$);
 4: if $m_1$ & $m_2 > m^{min}$ then
 5:     $[w, \gamma]$= PSVM($S_1, S_2, (a_1, ..., a_{N_a})$);
 6:     i=0;
 7:     while $i < m$ do
 8:         if $W * s_i - \gamma <= 0$ then
 9:             $s_i$ goes in the right node;
10:         else
11:             $s_i$ goes in the left node;
12:         end if
13:         $i \leftarrow i + 1$;
14:     end while
15: else
16:     $N$ is a leaf;
17: end if
Split_in_2_categories($S$)
Input: a training set $S$ with $N_c$ classes
Output: a training set $S_1$ and $S_2$ with 2 classes
 1: $[x_1, ..., x_{N_c}]$ = Number of training samples in each class
 2: Draw a class index to add to $S_1$ until $S_1$ contains at least $m/2$ samples.
Select_attributes($n$)
Input: the number of attributes $n$
Output: a set of attribute indices $a_1, ..., a_{N_a}$, with $N_a <= n$
 1: Draw a random number $N_a$ between 1 & $N_a^{max} < n$
 2: Draw $N_a$ attributes indices at random
PSVM($S_1, S_2, (a_1, ..., a_{N_a})$)
Input: the binary class training samples and the selected attributes
Output: the PSVM parameters $w$ & $\gamma$
```

also show the results given by the One Versus One linear SVM algorithm (OVO SVM), available in the OSU SVM Classifier Matlab Toolbox [2].

The number of trees is set to 100 for both PSVM trees and Extra-trees. Table 3 resumes the obtained results, m being the number of data samples, and n the number of attributes. The comparison of the PSVM Trees with the other methods shows a distinct improvement of the accuracies on the data sets where SVM based methods outperform the tree based method (i.e. the Vehicle and Isolet data sets). For the other data sets, our accuracies are close to the Extra-trees accuracies. Due to the computational efficiency of the PSVM algorithm, the computing time of the PSVM Trees training process is most of the time comparable to the Extra-trees, which has proven its attractive computational performance [5].

[2]http://www.kernel-machines.org/

Table 1: Training correctness with a 10-fold cross validation

Dataset $m \times n$, nb classes	linear MPSVM	OVO linear SVM	ET^d	**PSVM Trees**
Vowel, 528×10, 11 cl.	59.3 %	81.12 %	**98.26 %**	97.5 %
Vehicle, 846×18, 4 cl.	77.2 %	79.7 %	74 %	**84.02 %**
Segment, 2310×19, 7 cl.	91.0 %	93.37 %	**98.23 %**	97.89 %
Isolet, 7797×617, 26 cl.	93.8 %	95.9 %	92.39 %	**96.64 %**
Spambase, 4601×57, 2 cl.	92.2 %	92.89 %	95.83 %	**95.96 %**

4 Conclusion and Future Work

We proposed to embed (computationally efficient proximal) SVM in each node of an ensemble of randomized trees, to solve high-dimensional and multiple categories classification problems. Results showed that it outperformed in accuracy both the conventional ensemble of randomized trees algorithm and the proximal SVM methods on specific data sets while being close to the best accuracies on the others. This benefit comes from (1) the ability to take multiple features into account at each node of the trees; and (2) the ability to convert a multi-class problem into a hierarchy of binary classifications, without having to take a hard decision about the way to partition the multiple classes into binary sets. Future works could make a more detailed comparison on various datasets, with different types of classifiers. They could also investigate the benefit obtained from non linear kernels in the PSVM, or analyze how to trade-off randomization and optimal attributes selection at each node.

References

[1] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[2] L. Cheng, J. Zhang, J. Yang, and J. Ma. An improved hierarchical multi-class support vector machine with binary tree architecture. In *International Conference on Internet Computing in Science and Engineering (ICICSE'08)*, pages 106–109, Harbin, China, 2008.

[3] S. Cheong, S. Oh, and S.-Y. Lee. Support vector machines with binary tree architecture for multi-class classification. *Neural Information Processing Letters and Reviews*, 2(3):47–51, 2004.

[4] G. Fung and O. Mangasarian. Multicategory proximal support vector machine classifiers. *Machine Learning*, 59(1-2):pp. 77–97(21), 2005.

[5] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, April 2006.

[6] J.-Y. Lee and S. Olafsson. Multi-attribute decision trees and decision rules. *in Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques*, pages 327 – 358, 2006.

[7] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. *Advances in Neural Information Processing Systems*, 12:547 – 553, 2000.

[8] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Least squares support vector machine classifiers*, 9(3):293–300, 1999.

[9] V. N. Vapnik. The nature of statistical learning theory. In *Springer*, New York, USA, 1995.