Proceedings of International Joint Conference on Neural Networks, Atlanta, Georgia, USA, June 14-19, 2009

Creating an Ensemble of Diverse Support Vector Machines using Adaboost

Naiyan Hari Cândido Lima, Adrião Duarte Dória Neto, Jorge Dantas de Melo
Departament of Computer Engineering and Automation
Universidade Federal do Rio Grande do Norte
Email: {naiyan, adriao, jdmelo}@dca.ufrn.br

Abstract—**Support vector machines are one of the most employed methods of pattern classification, and the Adaboost algorithm is an effective way of improving the performance of the weak learners that compose the ensemble. In this article, we propose to create an Adaboost-based ensemble of SVM, by altering the Gaussian width parameter of the RBF-SVM. Using data sets from the UCI repository, we made tests to evaluate the algorithm.**

I. INTRODUCTION

In the recent years, researches on machine learning and pattern classification have been using two powerful established methods of achieving strong hypothesis, the Support Vector Machines (SVM) and the algorithm of adaptive boosting, Adaboost.

The SVM are statistical learning machines that are based on the principle of Structural Risk Minimization, developed by Vapnik [1]. With structures denominated kernels, the training set is mapped into a high-dimensional feature space, where it is easier for the optimization method to find the optimal separating hyperplane that maximizes the margin of separation between the two classes in a binary classification problem, using a regularization coefficient C to adjust between the training error and the complexity of the machine. From the common SVM kernel functions, we chose the RBF kernel, because of its Gaussian width parameter, σ, which provides greater control on the classification performance, given a suitable value of C [2]. Although SVM are stable and strong classifiers and are acknowledged to achieve good performances in most classification problems, the theoretically expected results are not always achieved.

Ensembles are techniques that obtain an output, for a classification or regression problem, combining the individual outputs of other learning models, aiming at an increase in the overall performance [3]. The most common ensemble methods are Bagging and Boosting, of which Boosting usually has superior results and Adaboost [4] (ADAptive BOOSTing) is the most popular boosting procedure. Adaboost receives a weight distribution on the training set, and modifies the weights based on the accuracy of the weak classifier. The poorly classified examples' weights are increased and the correctly classified examples' weights are decreased.

The training set for each classifier is sampled through the weight distribution at each round, and the examples that contribute the most to the general training error are more bound to compose the training set of the next round. Since the training examples are repeatedly shown to the classifier, Adaboost seems applicable when the size of the training set is diminutive.

However, there are evidences [5] that the application of boosting to classifiers like SVM might not work well compared with individual SVM and even sometimes cause a decline in the classification performance. However, as seen in [6], Adaboost requires a group of diverse classifiers to achieve good results, and the optimization in the SVM training reduces the diversity between two machines trained with the same input set.

We propose a simple way to ensure diversity by altering the Gaussian width parameter of the RBF kernel by scaling σ based on the accuracy of the previous iteration. Executing this modification, which we called σBoost, we achieved good performance compared with non-modified Adaboost implementation and with individual SVM.

This paper is organized as follows: Section II introduces the support vector machines and some of its characteristics, and the Adaboost algorithm. Section III describes the proposed method σBoost, while section IV has the experimental results and its analysis. The conclusion of the paper is presented on section V.

II. THEORETICAL BACKGROUND

A. Support Vector Machines

The SVM training is essentially the minimization of both the empirical error (training data error) and the structural error (generalization error), according to the Structural Risk Minimization principle [1]. The SRM is based on the fact that the generalization error rate is limited by the addition of the training error rate and a value that depends on the VC dimension [7], that is a measure of the capacity of a statistical classifier. Obtaining a balance between the two aforementioned error rates means overfitting avoidance, and good generalization rate. The support vectors define the boundaries of the class region, with the maximum margin attainable between the two classes.

Given the training set $(x_i, y_i)_{i=1}^N$ with input examples $x_i \in \mathbb{R}^n$ and the binary output $y_i \in \{-1, +1\}$, the SVM decision equation can be represented by:

March 9, 2009. This work was supported by CNPq - National Counsel of Technological and Scientific Development.

$$f(x) = \langle w, \phi(x) \rangle + b \quad (1)$$

where b is the bias parameter and $\phi(\cdot)$ is the non-linear mapping from the input space to the high-dimensional feature space and $\langle \cdot,\cdot \rangle$ represents the dot product in the feature space. To obtain a SVM with a soft margin, the training is essentially the minimization of the following function:

$$g(w, \xi) = \frac{1}{2} \|w^2\| + C \sum_{i=1}^{N} \xi_i \quad (2)$$

subject to the restriction:

$$y_i(\langle w, \phi(x) \rangle + b) \geq 1 - \xi_i \quad (3)$$

Where ξ_i is a non-negative parameter to relax the classification boundaries, classifying correctly examples that are shortly out of its class region and C is the regularization coefficient that establishes a balance between model complexity and training error. The optimization problem presented in (2) and (3) can also be expressed as:

$$\min W(\alpha) = -\sum_{i=1}^{N} \alpha_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j k(x_i, x_j) \quad (4)$$

subject to:

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \; and \; 0 \leq \alpha_i \leq C, i = 1, \ldots, N \quad (5)$$

where α_i is the Lagrange multiplier corresponding to the x_i sample, and $k(x_i, x_j)$ is the kernel function.

The kernel trick is used to define the $\phi(\cdot)$ non-linear mapping of the training samples. The most common kernel functions are the polynomial, the Gaussian RBF and 2-layer multilayer perceptron.

TABLE I – Common kernel functions for SVM

Kernel Function	Expression
Linear	$(x_i \cdot x_j)$
Polynomial	$(x_i \cdot x_j + 1)^p$
Gaussian RBF	$e^{-\|x_i - x_j\|^2 / 2\sigma^2}$
MLP	$\tanh(kx_i \cdot x_j - \delta)$

We chose the RBF kernel for our SVM ensemble. The SVM training method calculates the amount of centers in the RBF network kernel, its locations and weights, and the support vectors are the RBF centers in the input space. In [2] we see that the Gaussian width σ is more important in the definition of the general performance of the SVM than the regularization coefficient C, therefore it is σ our focal point in SVM customization.

B. Boosting and Adaboost

Boosting [8] is a machine-learning method that improves the performance of a given weak hypothesis (classifier whose performance is slightly better than random guessing) returning a strong classifier. Its motivation comes from the observation [9] that it is usually much easier finding a handful of inaccurate "rules of thumb" than one single reliable classification rule. Furthermore, since the PAC [10] – *Probably Approximately Correct* – learning model, and the ideas of *strong learnability* and *weak learnability*, there have been questions on the equivalence of the two notions.

Adaboost takes a set of training examples in the input domain X and handles a distribution over this data. The base classifier model, also called weak learner, is trained T times, one for each loop, and the accuracy of the current hypothesis defines the β parameter, used as the weight of the current classifier in the final hypothesis. Also using β, the distribution is updated, translating into a value of the difficulty of classification of the example. The distribution is used in the sampling, when some examples of the input set are selected to the training set of the base classifier at each loop. The harder it is to classify a said example, the more probability it has to be in the training set of the next round, and so the next classifiers focus on the mistakes of its predecessors. After the distribution is updated, a Z_t normalization factor scales the weights so that it still is a distribution. The final hypothesis is calculated by a linear combination of the weak hypothesis, its sign representing the predicted outcome of the ensemble for the input example.

The Adaboost pseudocode presented is for a binary classification problem. There are variations that accept multiple classes, but since SVM works better when applied to binary classification problems, we chose to develop over this version of Adaboost.

TABLE II – Adaboost

```
Given:
$S = \{(x_1, y_1), \ldots, (x_m, y_m); x_i \in X, y_i \in \{-1, 1\}\}$, the input training set
$T$ the number of iterations
Algorithm Adaboost:
    Initialize $D_1(t) = \frac{1}{m}, \; \forall (x_i, y_i) \in S$
    For $t = 1, \ldots, T$ do
        Train base classifier providing distribution $D_t$
        Obtain weak hypothesis $h_t : X \to \{-1, +1\}$
        Calculate $e_t = \sum_{i=1}^{m} D_1(t) \, |h_t(x_i) - y_i|$
        Calculate $\beta_t = \frac{e_t}{1 - e_t}$
        Update the distribution:
            $D_{t+1} = \frac{D_t(i)}{Z_t} \beta_t^{1 - |h_t(x_i) - y_i|}$
        Where $Z_t$ is the normalization factor.
    End For
    Output: Final Hypothesis:
        $H(x) = sign\left(\sum_{t=1}^{T} (log\,1/\beta_t) h_t(x) - \frac{1}{2} \sum_{t=1}^{T} log\,1/\beta_t\right)$
```

III. Proposed Method

Our main objective is to create an ensemble through the application of Adaboost on SVM. Initially we had implemented Adaboost, but, as expected, it did not improve consistently the performance of the weak classifier, as was attested in [5].

Proceeding, as the simple application of Adaboost did not achieve good results, we evaluated possible modifications on either SVM or Adaboost in view to an improvement in classification.

As said in [6], the accuracy of boosting is highly related to the diversity of the weak learners that compose the ensemble. In [2], we see that given an appropriate value for the SVM regularization parameter, C, the σ parameter of the RBF network kernel has a great effect on the RBF-SVM performance.

Thus, we propose an alteration on the Gaussian width of the weak learner, and expect that using RBF-SVM with different internal structures, we are going to acquire the growth in diversity that allows Adaboost to create an advanced and reliable classifying hypothesis.

The distribution issue is resolved by using it as a probability distribution, re-sampling the training set presented to the SVM at every round of the algorithm.

In order to apply such modification to the σ parameter of the RBF-SVM, we first had to evaluate a reasonable σ_{base}, the initial value of σ. One of such experiments' results is showed on Fig. 1. The σ_{base} value chose for the Iris dataset was 0.7, as it still yields a good, although not optimal performance, so that it can be boosted.

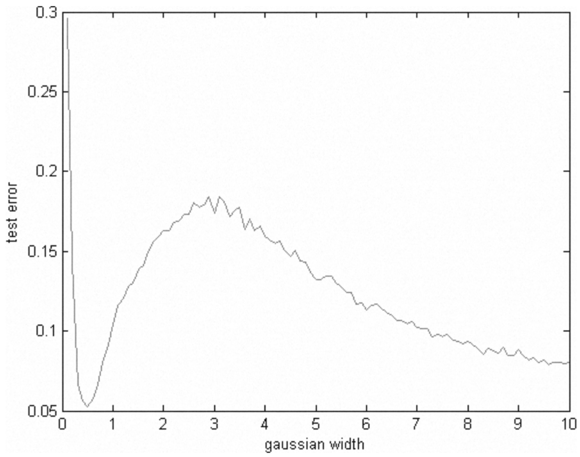

Figure 1 : Test error based on σ for SVM

The β value calculated round after round in the Adaboost routine is a measure of the importance of the weak classifier trained on that round in the final hypothesis of the ensemble, so, it is a good parameter in the adjustment of the internal RBF of the SVM. The more importance the current classifier has to the final hypothesis, the larger will be the difference between its kernel and the one of the next classifier.

So, the σBoost algorithm we propose changes the initial σ_{base} by adding it with a β scaled by a parameter we called σ_{scale}, that represents the level of influence of β on the structure of the RBF-SVM, and in which direction does the new σ value calculated does change, since it can assume both positive and negative values, σBoost+ or σBoost-.

TABLE III – σBoost

```
Given:
$S = \{(x_1, y_1), \dots, (x_m, y_m); x_i \in X, y_i \in \{-1, 1\}\}$, the input training set
$T$ the number of iterations
Algorithm σBoost:
    Initialize $D_1(t) = \frac{1}{m}, \forall (x_i, y_i) \in S$
    Set $\beta_0 = 0$
    For $t = 1, \dots, T$ do
        Set $\sigma = \sigma_{base} + (\sigma_{scale} * \beta_{t-1})$
        Train base classifier providing distribution $D_t$
        Obtain weak hypothesis $h_t: X \rightarrow \{-1, +1\}$
        Calculate $e_t = \sum_{i=1}^{m} D_1(t) \, |h_t(x_i) - y_i|$
        Calculate $\beta_t = \frac{e_t}{1 - e_t}$
        Update the distribution:
            $D_{t+1} = \frac{D_t(i)}{Z_t} \beta_t^{1 - |h_t(x_i) - y_i|}$
        Where $Z_t$ is the normalization factor.
    End For
    Output: Final Hypothesis:
        $H(x) = sign\left( \sum_{t=1}^{T} (log\, 1/\beta_t) h_t(x) - \frac{1}{2} \sum_{t=1}^{T} log\, 1/\beta_t \right)$
```

IV. Experimental results

The evaluation of the σBoost performance was made using nine data sets from the UCI repository [11]. We chose Abalone, Balance Scale, Congressional Voting Records, Contraceptive Method Choice, Ionosphere, Iris, Lenses, Promoter Gene Sequences and Wine, classification problems that were adjusted when needed, through either pairing or one-against-all approaches.

The Abalone problem is about predicting the age of the Abalone specimen through some of its characteristics. The Balance Scale problem is to see if a balance tips to the left, to the right or keeps equal, based on its weights and distances. The Congressional Voting Records problem is to predict the party of a given congressman based on his votes. The Contraceptive Method Choice problem is to predict whether a woman should employ a contraceptive method. The Ionosphere data set is composed of signals from high-frequency antennas targeting free electrons in the ionosphere. The Iris problem is to predict the species of a Iris family plant based on its petal and sepal information. The Lenses data set is composed of information of ophthalmologic patients to decide if they should use contact lenses. The Promoter Gene Sequences problem is to predict whether a gene sequence is of a promoter gene or not. The Wine problem is to classify between three classes of wine based on its characteristics.

Each data set was divided in a suitable amount of partitions, according to the amount of examples available, and for each partition we trained the classifier and calculated the test error with the other partitions, this process being repeated 10 times to generate a meaningful and representative set of results.

The size of the ensemble was 5 for most of the experiments. In the Wine data set, the value was 7 and for the Lenses, 10, and the regularization parameter C was arbitrarily large to minimize the number of misclassified points.

The following tables are a measurement of the average error percentage of the algorithms for each data set, and the standard deviation of said results. The tables also include the σ_{base} value used in each data set. The σ_{scale} applied was of 10% of the base value for σBoost+, and -10% for σBoost-. We assumed that a overly large σ_{scale} would undermine the influence of small σ_{base} values.

We compare the performance of both instances of σBoost with individual SVM trained at σ_{base} and non-modified Adaboost with SVM.

TABLE IV – Average Error %

Set	σ_{base}	SVM	Adaboost	σ-	σ+
Abalone	4.0	**25,49**	25,83	25,58	25,74
Balance	1.0	11,20	10,60	10,77	**10,04**
Congress	3.0	5,40	4,87	5,09	**4,71**
Contraceptive	1.5	36,79	34,58	**34,19**	34,63
Ionosphere	2.0	8,81	8,02	**7,44**	8,96
Iris	0.7	6,70	6,23	**5,20**	6,17
Lenses	3.0	28,99	29,72	29,86	**27,78**
Promoter	12.0	33,84	34,34	34,58	33,36
Wine	13.0	28,27	28,60	28,86	28,06

TABLE V – Standard Deviation

Set	σ_{base}	SVM	Adaboost	σ-	σ+
Abalone	4.0	0,18	0,31	0,24	0,19
Balance	1.0	0,57	0,60	0,71	0,84
Congress	3.0	0,72	0,39	0,20	0,31
Contraceptive	1.5	0,70	0,65	0,90	0,65
Ionosphere	2.0	0,95	1,16	0,80	0,87
Iris	0.7	1,58	1,47	1,21	1,63
Lenses	3.0	5,53	7,03	7,97	4,99
Promoter	12.0	2,80	2,84	2,14	1,61
Wine	13.0	1,49	1,26	1,56	1,42

As it can be perceived in the tables, in two of the data sets the results were too close to verify a difference in the performance of the methods, in one our methods were outperformed by SVM and in the others one of the instances of σBoost achieved superior results, being overall better than Adaboost.

However, since in some cases the error rate is not enough to measure the performance of a classifier, The ROC (Receiver Operating Characteristic) curve is one of the alternatives. It is a graphical plot of the true positive rate and the false positive rate of a classifier with the variation of its threshold. The area beneath the curve measures the performance of the classifier. More thresholds might be applied to get a smoother curve. We obtained the ROC curves for σBoost- and SVM using the Iris data set, as seen in Fig. 2.

Also, in comparison with other methods on the literature, we used the Splice Junction Gene Sequences, Breast Cancer and Thyroid Disease datasets from [12] [11] to compare our method with the results of SVM and Diverse AdaboostSVM in [2].

The Breast Cancer problem is to predict, based on nine characteristics, whether one subject has got breast cancer or not; The Splice Junction Gene Sequences data set is a set of DNA sequences, in which the classifier must recognize the boundaries between the parts of the DNA that are removed or not during the process of protein creation; and the Thyroid Disease data set has a record of thyroid diagnoses based on physical data of the subject.

Using the parameters C and σ_{base} obtained from [13], five rounds of boosting, and σ_{scale} of ±10%. The results are on Fig. 3.

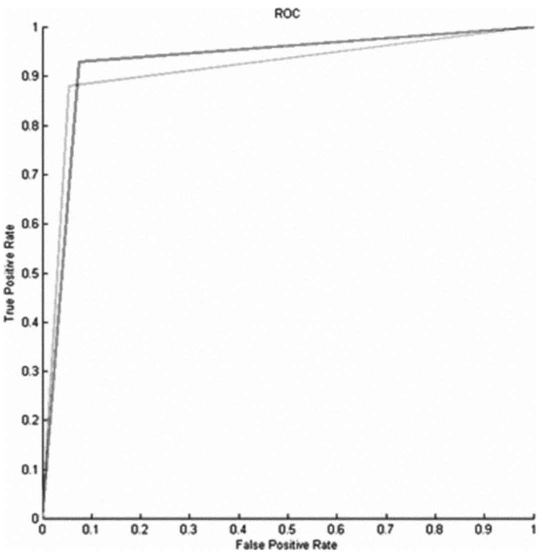

Fig. 2: ROC for the Iris data set in σBoost- (dark) and SVM (light)

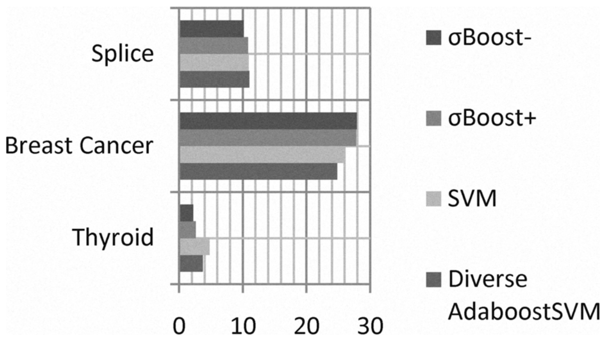

Fig. 3: Comparison between the classifiers' average error rate (%)

V. Conclusions

Our initial goal was to create an ensemble of support vector machines using Adaboost. As seen in the literature, Adaboost does not usually perform well, and that was our challenge.

Research brought the fact that the bad performance of the ensemble methods with strong classifiers could be circumvented by an increase in diversity.

The σBoost method, a slightly modified version of Adaboost that slightly alters the kernel of the SVM by changing the width of its internal Gaussian function, achieves better performance on the UCI data sets analyzed shown by its error rates and ROC curves, and has a simpler implementation than most Adaboost-based SVM ensemble algorithms already published.

In future works, we will try mixing other types of SVM kernels in the ensemble, expecting a gain in diversity, we intend to apply σBoost to other real-life problems, and to analyze the already obtained gain in performance to develop an even stronger classifier.

ACKNOWLEDGEMENTS

We would like to thank all the researchers of the LSI of UFRN for their support, Dr. Gunnar Rätsch for the data sets in the last round of tests.

REFERENCES

[1] Vladimir Vapnik, *Statistical Learning Theory*, John Wiley and Sons Inc., New York, 1998.

[2] Xuchun Li, Lei Wang, Eric Sung, *A Study of AdaBoost with SVM Based Weak Learners.* Proceedings of International Joint Conference on Neural Networks, 2005.

[3] Clodoaldo Aparecido de Moraes Lima. *Comitê de Máquinas: uma abordagem unificada empregando máquinas de vetores-suporte*. Universidade Estadual de Campinas, 2004.

[4] Yoav Freund and Robert Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of Computer and System Sciences, 1997.

[5] J. Wickramaratna, S. Holden, and B. Buxton apud Pedro Rangel, Fernando Lozano, and Elkin García, *Boosting of Support Vector Machines with application to editing*. Proceedings of the Fourth International Conference on Machine Learning and Applications, 2005.

[6] Kuncheva, L., Whitaker, C., 2003. *Measures of diversity in classifier ensembles and their relationship with ensemble accuracy*. Machine Learning 51 (2), 181{207.

[7] V. Vapnik and A. Chervonenkis. "On the uniform convergence of relative frequencies of events to their probabilities." *Theory of Probability and its Applications*, 1971.

[8] Robert E. Schapire. *The strength of weak learnability*. Machine Learning, vol. 5, no 2, pp. 197-227, 1990.

[9] Robert E. Schapire, *The boosting approach to machine learning: An overview*. MSRI Workshop on Nonlinear Estimation and Classification, 2002.

[10] Valiant L. G. apud Robert E. Schapire. *The strength of weak learnability*. Machine Learning, vol. 5, no 2, pp. 197-227, 1990.

[11] Asuncion, A. & Newman, D.J. (2007). UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.

[12] http://theoval.cmp.uea.ac.uk/~gcc/matlab/default.html#benchmarks

[13] http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm