ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

Constructing ensembles of classifiers using linear projections based on misclassified instances

César García-Osorio[1], and Nicolás García-Pedrajas[2] *

1-Department of Civil Engineering
University of Burgos,
Avda. Cantabria s/n, 09006 Burgos, Spain
email: cgosorio@ubu.es

2-Department of Computing and Numerical Analysis
University of Córdoba
Campus Universitario de Rabanales, 14071 Córdoba, Spain
e-mail: npedrajas@uco.es

Abstract. In this paper we propose a novel approach for ensemble construction based on the use of linear projections to achieve both accuracy and diversity of individual classifiers. The proposed approach uses the philosophy of boosting, putting more effort on difficult instances, but instead of learning the classifier on a biased distribution of the training set it uses misclassified instances to find a linear projection that favours their correct classification. Supervised linear projections are used to find the most suitable projection at each step of the creation of the ensemble. In a previous work we validated this approach using non-linear projections. In this work we show that linear projections can be used as well, with the advantage of being simpler, more interpretable and faster to obtain.

The method is compared with AdaBoost, showing an improved performance on a large set of 45 problems from the UCI Machine Learning Repository.

1 Introduction

An ensemble of classifiers consists of a combination of different classifiers, homogeneous or heterogeneous, to jointly perform a classification task [1]. A classification problem of K classes and n training observations consists of a set of instances whose class membership is known. Let $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots (\mathbf{x}_n, y_n)\}$ be a set of n training samples where each instance $\mathbf{x}_i$ belongs to a domain X. Each label is an integer from the set $Y = \{1, \ldots, L\}$. A multiclass classifier is a function $f : X \to Y$ that maps an instance $\mathbf{x} \in X \subset \mathbb{R}^D$ onto an element of Y.

The task is to find a definition for the unknown function, $f(\mathbf{x})$, given the set of training instances. In a classifier ensemble framework we have a set of classifiers $\mathbb{C} = \{C_1, C_2, \ldots, C_m\}$, each classifier performing a mapping of an instance vector $\mathbf{x} \in \mathbb{R}^D$ onto the set of labels $Y = \{1, \ldots, L\}$.

Boosting methods are the most popular techniques for constructing ensembles of classifiers. Their popularity is mainly due to the success of AdaBoost.

*This work has been partially funded by the project BU004B06 from the Consejería de Educación de la Junta de Castilla y León (Spain).

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

Boosting constructs an ensemble in a stepwise manner. At each step a new classifier is added to the ensemble. The basic idea is that the new classifier is trained on a distribution of the learning instances biased towards the most difficult instances. In this way, each instance has an associated weight that is higher if the instance has been misclassified by several of the previous classifiers. AdaBoost tends to perform very well for some problems but can also perform very poorly on other problems. One of the sources of the bad behaviour of AdaBoost is that although it is always able to construct diverse ensembles, in some problems the individual classifiers tend to have large training errors. Moreover, AdaBoost usually performs poorly on noisy problems [2].

One of the sources of failure of boosting is putting to much stress on correctly classifying all the instances. Outliers or noisy instances become too relevant in the training set undermining the performance of the ensemble. In a previous work [1] we constructed ensembles projecting the input variables in a way that made easier the classification of misclassified instances. This projection was performed using the hidden layer of a multilayer perceptron. In this paper we show how we can use supervised linear projection methods to perform the same task in an easier and faster way.

This approach is able to incorporate the advantages of boosting without its main drawbacks. The construction of the projection taking into account only instances that have been misclassified by a previous classifier permits the new classifier to focus on difficult instances. Nevertheless, as this classifier receives a uniform distribution of the training instances, the sensitivity to noise and the effect of small datasets is greatly reduced. The proposed method at each step t considers only the subset of instances, $S' \subset S$, misclassified by the classifier added in step $t-1$. It uses the instances in S' to obtain a linear projection that is focused only on misclassified instances. The proposed method is shown in Algorithm 1. The next section explains how the supervised linear projections are obtained.

Data : A training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, a base learning algorithm, $\mathbb{L}$, and the number of iterations T.

Result : The final classifier: $C^*(\mathbf{x}) = \arg\max_{y \in Y} \sum_{t:C(\mathbf{x})=y} 1$.

1 $C_0 = \mathbb{L}(S)$
for $t = 1$ *to* $T-1$ **do**
2 $S' \subset S$, $S' = \{\mathbf{x}_i \in S : C_{t-1}(\mathbf{x}_i) \neq y_i\}$.
3 Obtain supervised linear projection $\mathbf{P}(\mathbf{x})$ using S'.
4 $C_t = \mathbb{L}(\mathbf{P}(S))$
end

Algorithm 1: Linear Projection Boosting algorithm.

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

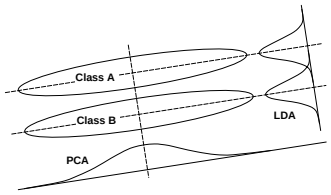

Fig. 1: PCA fails when the class labels are not used.

2 Linear Supervised Projections

One of the most used methods for linear projection is Principal Component Analysis (PCA). PCA projects the data set onto the directions which explain most of the variance in the data set. Assuming Gaussian distribution, these are the directions with more information. For our methodology, the main drawback of PCA is that it is an unsupervised technique and does not take into account the class labels of the data set. PCA is more a method for efficient representation rather than a method for efficient discrimination. A typical example where PCA fails is shown in Figure 1. In such cases we need a supervised technique as Linear Discriminant Analysis.

2.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) was first used for two classes. It finds a linear subspace that maximises the class separability. The objective is to find a projection W that maximises the ratio of between-class scatter S_b against within-class scatter S_w [3]. LDA has two important disadvantages: i) Gaussian assumption over the class distribution of the data samples; and ii) the dimensionality of the subspaces obtained is limited by the number of classes; for a L classes data set, at most $L-1$ dimensional.

2.2 Nonparametric Discriminant Analysis

Nonparametric Discriminant Analysis (NDA) is an alternative to LDA that avoids its limitations. Fukunaga and Mantock [3] presented this nonparametric technique illustrating it for the two class case. The formulation for the multiclass case is as follows [4].

$$S_b^{\mathrm{NDA}} = \sum_{i=1}^{L} P(\omega_i) \sum_{\substack{j=1 \\ j\neq i}}^{L} \sum_{l=1}^{N_i} \frac{w_l^{(i,j)}}{N_i} \left(x_l^{(i)} - M_j^k(x_l^{(i)})\right) \left(x_l^{(i)} - M_j^k(x_l^{(i)})\right)^T \quad (1)$$

$$S_w^{\mathrm{NDA}} = \sum_{i=1}^{L} P(\omega_i) \sum_{l=1}^{N_i} \frac{w_l^{(i,i)}}{N_i} \left(x_l^{(i)} - M_i^k(x_l^{(i)})\right) \left(x_l^{(i)} - M_i^k(x_l^{(i)})\right)^T \quad (2)$$

$$w_l^{(i,j)} = \frac{\min\left\{d^\alpha\left(x_l^{(i)}, x_{\text{kNN}}^{(i)}\right), d^\alpha\left(x_l^{(i)}, x_{\text{kNN}}^{(j)}\right)\right\}}{d^\alpha\left(x_l^{(i)}, x_{\text{kNN}}^{(i)}\right) + d^\alpha\left(x_l^{(i)}, x_{\text{kNN}}^{(j)}\right)} \tag{3}$$

where α is a control parameter between zero and infinity, N_i is the number of instances in class i, $d(x_l^{(i)}, x_{\text{kNN}}^{(j)})$ is the distance from $x_l^{(i)}$ in class i to its k-th nearest neighbour (NN) in class j and $M_j^k(x_l^{(i)}) = (1/k)\sum_{t=1}^{k} x_{\text{tNN}}^{(j)}$ is the mean vector of the k nearest neighbour of $x_l^{(i)}$ in class j.

NDA has two disadvantages: i) Parameters k and α are usually decided by rules of thumb. So the best result usually comes after several trials; and ii) when the within-class scatter matrix in NDA is still in parametric form and the training set size is small, NDA will have the singularity problem.

2.3 Regularisation

Since, in the previous methods, within scatter matrices have to be inverted in order to calculate the linear projection, it is important that these matrices are not singular (problem *ill-posed*) or close to singular (problem *poorly posed*). This is usually the case for small data sizes and high dimensionality.

To overcome these problems we can use regularisation. In the experiments, as a rule of thumb, whenever the size of the data set is smaller than four times the dimension, we have applied the regularisation process proposed in [5]. The new regularised version of within-class scatter matrix is

$$S_w^R = \alpha\text{diag}(S_w) + \beta\frac{\text{trace}(S_w)}{d}I + \gamma S_w \tag{4}$$

where d is the dimensionality of data, α, β and γ are mixing parameters with $0 \leq \alpha, \beta, \gamma \leq 1$ and $\alpha + \beta + \gamma = 1$. In the experiments reported in this paper $\alpha = \beta = \gamma = 0.33$.

2.4 Hybrid Discriminant Analysis

Tian et al. present in [6] the Hybrid Discriminant Analysis (HDA) as a framework that unifies PCA and LDA. The ratio to maximise in HDA is

$$\arg\max_W \frac{\left|W^T\left((1-\lambda)S_b + \lambda\widehat{\Sigma}\right)W\right|}{\left|W^T\left((1-\eta)S_w + \eta I\right)W\right|} \tag{5}$$

where I the identity matrix and $\widehat{\Sigma}$ the covariance matrix. The combination of values $(\lambda = 0, \eta = 0)$ gives LDA. PCA can be obtained with values $(\lambda = 1, \eta = 1)$. For other values, equation (5) provides a set of alternatives between PCA and LDA.

The main advantages of the method are: i) With values $\lambda \neq 0$ the matrix $(1-\lambda)S_b + \lambda\widehat{\Sigma}$ if full rank and HDA overcomes one of the limitations of LDA and we are now not restricted to projections of $L-1$ dimensions at most; and

ii) with values $\eta \neq 0$ we get a simple regularisation scheme which avoids the singularity of $(1-\eta)S_w + \eta I$.

3 Experimental setup

For the assessment of the validity or our method we selected 45 datasets from the UCI Machine Learning Repository. The experiments were conducted following the 5x2 cross-validation set-up. Demšar [7] proposed several methodologies to make comparisons among several methods. Following this paper we carry out pairwise comparisons using a Wilcoxon test. This test is recommended because it was found to be the best one for comparing pairs of algorithms [7]. In the experiments we test AdaBoost algorithm against our approach, so our comparisons are always paired.

4 Experiments

It is clear that the selection of the base learner may have a significant effect on the results of the proposed method. Thus, we have studied the proposed model using three different base learners: a multilayer neural network trained using the standard back-propagation algorithm, the C4.5 algorithm, and a Support Vector Machine with a Gaussian kernel.[1] Table 1 shows the comparison of the two methods based on linear projections (HDA and NDA) and AdaBoost.

	C4.5		Neural nets		SVM	
	HDA	NDA	HDA	NDA	HDA	NDA
AdaBoost	31/14	30/15	30/15	29/16	29/16	31/13
	0.0650	0.0405	0.0439	0.0633	0.0198	0.0007

Table 1: Comparison of AdaBoost and the proposed methods in terms of testing error. Win/loss record and the p-value of Wilcoxon test are shown.

The results are illustrated in Figure 2. The figure represents for each point the testing error of the standard method and ours. Points below the diagonal line show a better performance of our method, and points above the diagonal line show a better performance of AdaBoost. We can see that there are more points below the diagonal, and also that the separation of these points from the diagonal is larger. Table 1 shows that the differences are significant at a confidence level of 90% for all the classifiers, and at a 95% for HDA for neural networks and SVMs and for NDA for C4.5 and SVMs.

5 Conclusions

In this paper we have shown how we can construct ensembles of classifiers by means of linear projections that are obtained using misclassified instances. The

[1] The SVM learning algorithm was programmed using functions from the LIBSVM library.

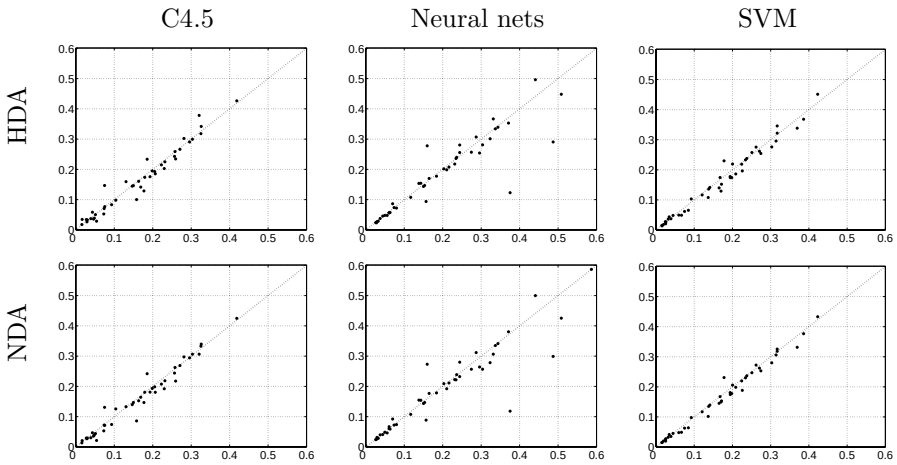

Fig. 2: Comparison of testing error for AdaBoost method (x-axis) and the proposed methodology (y-axis).

proposed methodology is able to significantly improve the performance of AdaBoost algorithm in a large set of 45 problems and three different base learners.

As future research line, we are working on using the weights of the instances given by boosting to construct the linear projections, instead of using only the miss-classified instances. Another interesting line in which we are interested is trying this technique for regression.

References

[1] N. García-Pedrajas, C. García-Osorio, and C. Fyfe. Nonlinear boosting projections for ensemble construction. *Journal of Machine Learning Research*, 8:1–33, 2007. (Cites: 3).

[2] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1/2):105–142, July/August 1999.

[3] K. Fukunaga and J. Mantock. Nonparametric discriminant analysis. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 5(6):671–678, 1983.

[4] B-C. Kuo and D. A. Landgrebe. Nonparametric weighted feature extraction for classification. *IEEE Transactions on Geoscience and Remote Sensing*, 42(5):1096–1105, May 2004.

[5] B.-C. Kuo, L.-W. Ko, C.-H. Pai, and D. A. Landgrebe. Regularized feature extractions for hyperspectral data classification. In *International Geoscience and Remote Sensing Symposium*, volume 3, pages 1767–1769, USA, July 2003.

[6] Qi Tian, Jie Yu, and Thomas S. Huang. Boosting multiple classifiers constructed by hybrid discriminant analysis. In Nikunj C. Oza, Robi Polikar, Josef Kittler, and Fabio Roli, editors, *Multiple Classifier Systems, 6th International Workshop, MCS*, volume 3541 of *Lecture Notes in Computer Science*, pages 42–52, Seaside, CA, USA, June 2005. Springer.

[7] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.