Exploiting Diversity of Margin–based Classifiers

Enrique Romero, Xavier Carreras and Lluís Màrquez
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
{eromero,carreras,lluism}@lsi.upc.es

***Abstract*— An experimental comparison among Support Vector Machines, AdaBoost and a recently proposed model for maximizing the margin with Feed–forward Neural Networks has been made on a real–world classification problem, namely Text Categorization. The results obtained when comparing their agreement on the predictions show that similar performance does not imply similar predictions, suggesting that different models can be combined to obtain better performance. As a consequence of the study, we derived a very simple confidence measure of the prediction of the tested margin–based classifiers. This measure is based on the margin curve. The combination of margin–based classifiers with this confidence measure lead to a marked improvement on the performance of the system, when combined with several well–known combination schemes.**

I. Introduction

Recent years have seen the explosion of margin–based classification systems in a wide variety of applications. Typically, the inductive bias of a margin–based learning method allows to consider the output values of a classifier as a good measure of the confidence on its predictions. In this work, several comparisons among Support Vector Machines (SVM) [15], AdaBoost [13] and a recently proposed model for maximizing the margin with Feed–forward Neural Networks (FNN) [9] have been made. The empirical study has been performed on the Text Categorization task, a real–world classification problem from the Information Retrieval domain, and extends initial experimentation involving the aforementioned algorithms [10]. To the best of our knowledge, this is the first so detailed empirical comparison made among margin–based classifiers. The evidence that there exist important differences in the predictions of several models with good performance suggests that they can be combined in order to obtain better results than every individual model. For this purpose, we studied a way to assign to every prediction a confidence factor depending on its output value. Although the ranges of the margins are very different for different models, the shape of the margins curves in the training and test sets were very similar among them. Therefore, we can measure the confidence of every prediction of a classifier taking its margin curve as a reference. This value can be computed as the ratio between the position in the distribution and the total number of predictions in the reference distribution. The combination of margin–based classifiers with heuristics based on these ideas lead to a marked improvement on the performance of the system in a consistent way, when combined with several well–known combination schemes.

The overall organization of the paper is as follows. The description of the margin–based classifiers tested can be found in section II. The experimental work is described in section III. In section IV, the comparison among the learned models is discussed. Section V is devoted to describe the construction of the confidence measure for these margin–based classifiers. Finally, section VI concludes and outlines some directions for further research.

II. Margin–based Classifiers

This section briefly describes the margin–based models compared in the experiments.

Support Vector Machines. According to [15], SVM can be described as follows: the input vectors are mapped into a (usually high–dimensional) inner product space through some non–linear mapping ϕ, chosen *a priori*. In this space (the *feature space*), an optimal separating hyperplane is constructed. In SVM, an optimal separating hyperplane means a hyperplane with maximal distance with respect to the closest example in the training set (maximal normalized margin). The (functional) margin of a point (x_i, y_i) with respect to a function f is defined as $\mathrm{mrg}(x_i, y_i, f) = y_i f(x_i)$. By using a kernel function $K(u, v)$ the mapping can be implicit, since the inner product defining the hyperplane can be evaluated as $\langle \phi(u), \phi(v) \rangle = K(u, v)$ for every two vectors $u, v \in \mathbb{R}^N$. The most usual kernel functions $K(u, v)$ are polynomial, Gaussian–like or some particular sigmoids.

Margin maximization, derived from statistical learning theory, has been proved to be a good inductive bias, both theoretically and in a wide variety of practical applications [4].

AdaBoost. The purpose of boosting algorithms is to find a highly accurate classification rule by combining many *weak* or *base* classifiers. In this work we use the generalized AdaBoost algorithm presented in [13] by Schapire and Singer.

Let $(x_1, y_1), \ldots, (x_m, y_m)$ be the set of m training examples, where each x_i belongs to an input space $\mathcal{X}$ and $y_i \in \mathcal{Y} = \{+1, -1\}$ is the corresponding class label. AdaBoost learns a number T of base classifiers, each time presenting the base learning algorithm a different weighting over the examples. A base classifier is seen as a real–valued function $h : \mathcal{X} \to \mathbb{R}$. The output of each h_t is a real number whose sign is interpreted as the predicted class, and whose magnitude is the confidence in the prediction. The AdaBoost classifier is a weighted vote of the base classifiers, given by the expression $f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$, where α_t represents the weight of h_t inside the whole classifier. Again, the sign of

$f(x)$ is the class of the prediction and the magnitude is its confidence.

The learning bias of AdaBoost is proven to be very aggressive at maximizing the margin of the training examples and this makes a clear connection to the SVM learning paradigm [13]. More details about the relation between AdaBoost and SVM can be found in [8], [12].

The base classifiers we use are decision trees of fixed depth. The internal nodes of a decision tree test the value of Boolean predicate (e.g. "the word *dollar* occurs in the document"). The leaves of a tree define a partition over the input space $\mathcal{X}$, and each leaf contains the prediction of the tree for the corresponding part of $\mathcal{X}$. We follow the criterion presented in [13] for growing base decision trees and computing the predictions in the leaves. A maximum depth is used as the stopping criterion.

Feed–forward Neural Networks for Margin Maximization. A margin–based learning model for FNN is presented in [9]. The key idea of the model is a weighting of the sum–of–squares error function, inspired by the AdaBoost algorithm. This weighting function modifies the contribution of every point to the total error depending on its margin. The proposed weighting function is

$$D(x_i, y_i, \alpha^+, \alpha^-) = \begin{cases} e^{-|mrg|^{\alpha^+}} & \text{if } mrg \geqslant 0 \\ e^{+|mrg|^{\alpha^-}} & \text{if } mrg < 0 \text{ and } \alpha^- \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

where the margin $mrg = \mathrm{mrg}(x_i, y_i, f) = y_i f(x_i)$ as usual. In the linearly separable case, the hyperplane that maximizes the normalized margin also minimizes asymptotically the weighted sum–of–squares error function proposed. The hardness of the margin can be controlled, as in SVM, so that this model can be used for the non–linearly separable case as well (see [9], [10] for details).

All experiments conducted with this model were performed with standard Back–propagation (BP) [11] weighting the sum–of–squares error derivative, as in [10]. Every architecture had linear output units. From now on, we will refer to this method as BPW.

III. Experimental Setting

We have concentrated on a classical problem from the Information Retrieval domain, namely Text Categorization (TC), for carrying out the experimental evaluation. Text categorization, or classification, is the problem of automatically assigning text documents to a set of pre–specified categories, based on their contents. From the 90's, many statistical and machine learning algorithms have been applied to the TC task, including among others: rule induction, decision trees, bayesian classifiers, neural networks, on–line linear classifiers, instance–based learning, boosting–based committees, SVM, and regression models. There is a general agreement in that SVM and boosting–based committees are among the top–notch performance systems [14] in this task.

Data Set. We have used the publicly available Reuters-21578 collection of documents[1], which can be considered the most important benchmark corpus for the TC task. This corpus contains $12,902$ documents of an average length of about 200 words, and it is divided (according to the "ModApte" split) into a training set of $9,603$ examples and a test set of $3,299$ examples. The corpus is labeled using 118 different categories and has a ratio of 1.2 categories per document. However, the frequency distribution of these categories is very extreme (the 10 most frequent categories covers 75% of the training corpus, and there are 31 categories with only one or two examples). For that reason, we have considered, as in many other works, only the 10 most frequent categories of the corpus. As a consequence, our training corpus contains $3,113$ documents with no category and a ratio of 1.11 categories per document in the rest. Table I shows the number of examples for every category.

Features. Regarding the representation of the documents, we have used the simplest *bag-of-words* model, in which each feature corresponds to a single word and all features are binary valued, indicating the presence or absence of the words in the documents. We discarded using more complex document representations or feature weighting schemes since the main goal of this paper is not to achieve the best results on the TC task, but to make comparisons among several models in a simple and controlled framework. The attributes have been filtered out by selecting the 50 most relevant for each of the ten classes and merging them all in a unique feature set, containing 387 features. The relevance measure used for ranking attributes is the RLM entropy–based distance function used for feature selection in decision–tree induction [6].

Evaluation Measures. TC is a multiclass multilabel classification problem, since each document may be assigned a set of categories. Thus, one may think that a *yes/no* decision must be taken for each pair $(document, category)$, in order to assign categories to the documents. The most standard way of evaluating TC systems is in terms of *precision* (P), *recall* (R), and the F_1 measure. Precision is defined as the ratio between the number of correctly assigned categories and the total number of categories assigned by the system. Recall is defined as the ratio between the number of correctly assigned categories and the total number of real categories assigned to examples. The F_1 measure is the harmonic mean of precision and recall: $F_1(P, R) = 2PR/(P + R)$.

Models. The classification models tested can be seen in the first two columns of table II. We used the LIBSVM software [2][2] to test several models with linear (*lin*) and gaussian (*gau*) kernels. As usual, the problem was binarized for SVM. Similarly to SVM, AdaBoost also needed the binarization of the data set. We trained 6 different AdaBoost models (generalized AdaBoost algorithm with confidence–rated predictions, as described in [13]) by varying the complexity of the weak rules from decision stumps to decision trees of depth 5. Several

[1] Available at www.daviddlewis.com/resources/testcollections.

[2] Freely available from www.csie.ntu.edu.tw/∼cjlin/libsvm.

TABLE I

NUMBER OF EXAMPLES FOR THE 10 MOST FREQUENT CATEGORIES IN THE TC PROBLEM FOR THE TRAINING AND TEST SETS

	earn	acq	money	grain	crude	trade	interest	wheat	ship	corn	None
Training Set	2,877	1,650	538	433	389	369	347	212	197	181	3,113
Test Set	1,087	719	179	149	189	117	131	71	89	56	754

Multi–layer Perceptrons (MLP) architectures were trained with BPW and different number of hidden units and activation functions: linear (*lin*), hyperbolic tangent (*tnh*) and sine (*sin*). Output activation functions were always linear. The F_1 results for BPW are always the average–output committee of the resulting networks for 5 different runs.

Model Selection. In order to conduct a fair experiment, model selection was performed on the training set. In doing so, a 5–fold cross–validation (CV) experiment was performed, and the parameters that maximized accuracy were selected for training the final classifiers using the whole training set.

Table II contains the best parameterizations according to the model selection and the F_1 results obtained by the corresponding classifiers on the 5–fold CV and the test corpus, micro–averaged over the 10 categories. The F_1 results achieved on the task are competitive, given the simple document representation used. As an example, a recent work [5] shows performances between 87.5 and 92.0 on the same data set, by using linear SVM with many sophisticated feature weightings.

IV. COMPARING MARGIN–BASED CLASSIFIERS

Comparing the Predictions on the Test Set. More insight on the learned models can be obtained by comparing the partitions that every model induced on the input space, rather than solely the accuracy values achieved. For that, we calculated the agreement ratio between each pair of models on the test set (i.e., the proportion of test examples in which the two classifiers agree in their predictions). Additionally, we calculated the Kappa statistic (κ). The Kappa statistic is a measure of inter–annotator agreement which reduces the effect of chance agreement [3]. It has been used for measuring inter–annotator agreement during the construction of some semantic annotated corpora [7]. A Kappa value of 0 indicates that the agreement is purely due to chance agreement, whereas a Kappa value of 1 indicates perfect agreement. A Kappa value of 0.8 and above is considered as indicating good agreement. Both the agreement ratio and the Kappa statistic have been computed without taking into account the well classified negative examples, since these predictions are not relevant at all for the task (note that by default an example does not belong to any category). Indeed, well classified negative examples are neither considered in the F_1 measure.

Table III contains a subset of these comparisons which allows us to extract some interesting conclusions about the similarities and differences among the models learned:

- Linear models are more similar among them than non–linear ones. This effect is also observed for AdaBoost models with simplest weak hypotheses.
- SVM models are very similar among them.
- None of the AdaBoost models is very similar to either BPW or SVM models.
- Although it could be expected that AdaBoost models could be very similar among them, we observed that it was not the case, since the maximum similarity found was between *ab-depth4* and *ab-depth5*, with an agreement rate of 92.67% and a Kappa value of 0.70. Models *ab-stumps* and *ab-depth1*, for example, did not agree more than 87.67% (and a Kappa value of 0.53) with any other AdaBoost model. For the sake of simplicity, these results are not included in the tables.

These results show that the performance of the obtained models seems to be independent of the similarities among them, whatever the learning model is used, i.e., there exist SVM, AdaBoost and BPW classifiers with a good performance and different behaviors on the test set. This observation opens the avenue to combine classifiers in this problem.

Comparing the Margin Curves and Relative Margins. In order to look for a criterion to combine the learned classifiers, we compared their margin curves on the test set. It can be observed that, whereas the ranges of the margins are quite different, the shape of the margins curves are very similar for all the models. As an example, margin curves for models *svm-gau*, *ab-depth3* and *bpw-tnh-lin* are shown in figure 1 for the binarized problem of class *earn* (the most frequent category). In the X axis of the plots we have the examples ordered by its margin (i.e., the first points in the left of each plot are those training examples with a lower margin value). In the Y axis the margin of the example is plotted. It can be observed that misclassified examples have relatively small (in absolute value) negative margins, whereas points with large values always correspond to well classified examples. Note the different ranges of margins of the respective classifiers.

Since the number of examples with margin near to 0 is small, the margin curves alone cannot explain the observed differences among the models (see table III). A possible explanation is that there exist examples with very different margins in different models, but this effect is compensated among examples to have similar margin curves shapes. The relative margin of the examples of models in figure 1 was compared, confirming this claim. The results of the comparison can be seen in figure 2. The X axis plots the examples of every respective model, ordered by its margin value, as in figure 1. The Y axis shows the position that each example would occupy if the model in the Y axis had been ordered following the same criterion (the relative margin). A straight line ($Y = X$)

TABLE II

PARAMETERS SELECTED BY THE MODEL SELECTION PROCEDURE AND F_1 RESULTS OBTAINED BY THE CORRESPONDING SVM, ADABOOST AND BPW CLASSIFIERS IN THE 5–FOLD CV AND THE TEST SET

Identifier	**Software**	**C–value**	F_1 (5–fold CV)	F_1 (test)
svm-lin	LIBSVM	70	87.48	89.05
svm-gau	LIBSVM	30	87.68	89.36
Identifier	**Algorithm**	**Rounds**	F_1 (5–fold CV)	F_1 (test)
ab-stumps	AdaBoost	200	86.35	87.92
ab-depth1	AdaBoost	100	87.09	88.63
ab-depth2	AdaBoost	300	87.29	88.78
ab-depth3	AdaBoost	300	87.21	89.01
ab-depth4	AdaBoost	500	87.34	88.50
ab-depth5	AdaBoost	500	87.21	88.97
Identifier	**Algorithm**	**Epochs**	F_1 (5–fold CV)	F_1 (test)
bpw-lin	BPW	140	87.45	89.12
bpw-tnh-lin	BPW (35 hidden)	110	88.38	89.96
bpw-sin-lin	BPW (20 hidden)	60	88.19	89.84

TABLE III

PERCENTAGES OF AGREEMENT (AGR) AND KAPPA (KAP) VALUES AMONG SEVERAL SVM, ADABOOST AND BPW MODELS ON THE TEST SET

	svm-lin		*svm-gau*		*bpw-lin*		*bpw-tnh-lin*		*bpw-sin-lin*	
	Agr	Kap	Agr	Kap	Agr	Kap	Agr	Kap	Agr	Kap
svm-lin	–	–	98.39	0.93	96.05	0.83	92.43	0.68	93.10	0.71
svm-gau	98.39	0.93	–	–	96.31	0.84	93.27	0.71	94.13	0.75
ab-stumps	90.66	0.65	90.84	0.65	91.54	0.67	90.62	0.62	90.25	0.61
ab-depth1	87.44	0.52	88.13	0.54	88.12	0.53	89.34	0.56	89.15	0.56
ab-depth2	86.11	0.47	86.86	0.49	86.79	0.49	88.27	0.52	87.95	0.51
ab-depth3	86.97	0.50	87.52	0.52	87.53	0.52	88.54	0.53	88.47	0.53
ab-depth4	86.48	0.49	87.29	0.51	87.30	0.51	88.07	0.52	88.20	0.53
ab-depth5	86.18	0.48	87.06	0.50	86.99	0.50	88.07	0.51	88.01	0.52

would indicate an exact coincidence between the two models. Clearly, there exists a very strong correlation between *svm-gau* and *bpw-lin* with regard to the relative margin, whereas the other models are less correlated. Therefore, these models are not only similar or different in their predictions, but also in the importance that both give to the examples in the data set.

Similar results to those presented in figures 1 and 2 were also observed for the remaining nine categories of the problem and data sets (the training set and the training and test sets of the 5–fold CV).

V. EXPLOITING DIVERSITY OF MARGIN-BASED CLASSIFIERS

In the previous section, we have seen that quite different classification models may have similar performances, with very similar margin curves. This fact allows to scale the predictions of every classifier in order to obtain a confidence value within a fixed and normalized range, under the same criterion for all classifiers. This is an important issue, since a confidence measure has to be independent of the particular inductive bias that different learning algorithms use. These confidence values will be used to combine the predictions of the classifiers.

The procedure works as follows. First, we obtain a reference margin–based distribution for every model and every class. Second, we compute the position that every prediction occupies in the reference distribution (i.e., the number of predictions in the distribution with lower value than the current one). The confidence value of every prediction is computed as the ratio between the position in the distribution and the total number of predictions in the reference distribution. Note that the output values cannot be directly used because the ranges of the margins were quite different (see figure 1).

In order to obtain the reference distributions, for every model and every class we take the predictions in the test sets of the 5–fold CV performed in the model selection step[3]. Due to the different frequencies of positive and negative examples in the data set, we consider a reference distribution for

[3]Observe that this distribution would be the margin distribution if all the examples were correctly classified

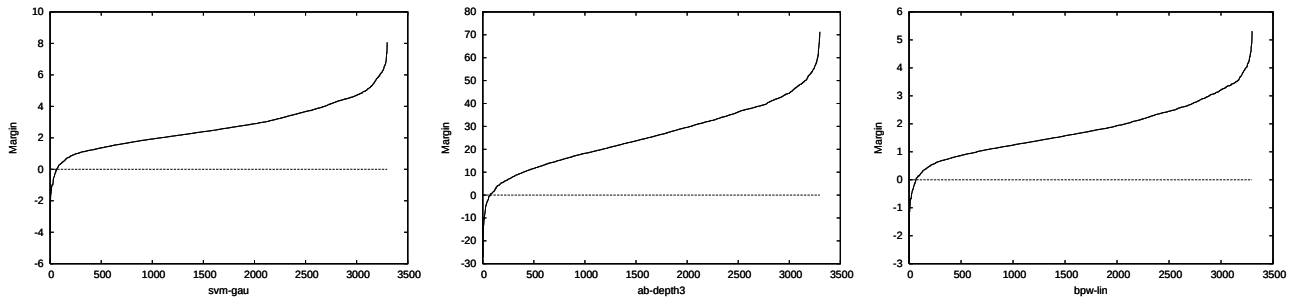

Fig. 1. Comparison of the margin curves among *svm-gau* (left), *ab-depth3* (middle) and *bpw-lin* (right) on the test set for the most frequent class

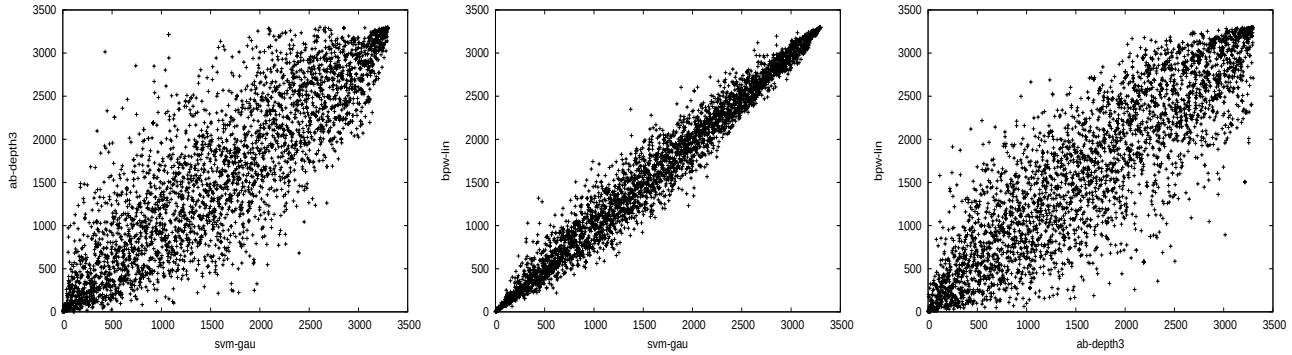

Fig. 2. Comparison of the relative margin of *svm-gau* and *ab-depth3* (left), *svm-gau* and *bpw-lin* (middle), and *ab-depth3* and *bpw-lin* (right) on the test set for the most frequent class

positive predictions and another one for negative predictions. Therefore, the total number of reference distributions is $2MC$, where M is the number of models and C is the number of classes. Given a prediction of a classifier on a test example, its reference distribution is either the positive or the negative one depending on the sign of that prediction. Then, its confidence value is the percentage of predictions (of the same model, class and sign) which are, in absolute value, lower than the current one. Note that these confidence values can be computed for every classifier in an independent way (i.e., only the predictions of that classifier are needed).

The procedure previously described was used to compute the confidence of every prediction for every model described in section III. In order to test these confidence values, we performed several combination experiments with three combination schemes. These experiments were carried out firstly without the confidence values of the predictions, and secondly with these values. For every combination scheme, the resulting ensemble of classifiers was determined on the training set (within the 5–fold CV setting) by a greedy procedure that departs from the best model (*bpw-tnh-lin*) and then iteratively adds the classifier that maximizes the increase of the F_1 measure.

In a first step, we combined the learned models without using the confidence measure previously described. We used three well–known combination schemes: Majority Voting (MV), Weighted Voting (WV) and Pairwise Voting (PV). In the MV scheme, given a fixed category and an example, each system predicts whether this category should be assigned to the example or not. The majority option is selected. In WV, each classifier votes with a weight proportional to its F_1 score estimated in the model selection step. The PV scheme is a little more complex and powerful since, in principle, it might recover a category that receives a minority of the individual votes or even none of them. Let $y^i \in \{-1, +1\}$ be the prediction of classifier C_k^i, on a class k and example x. Given an example and category pair, the PV scheme calculates $PV(x,k) = \arg\max_{y \in \{-1,+1\}} \sum_{i \neq j} P(y \mid y^i \wedge y^j)$. The probabilities of y given the predictions of each pair of classifiers are estimated by maximum likelihood from frequency counts on the model selection step. For non observed pairs of predictions (y^i, y^j) we back–off to conditional probabilities on the predictions of individual classifiers. Similar to the computation of the proposed confidence values, the results on the test sets of the 5–fold CV performed in the model selection step were used as the reference to obtain any needed information (the F_1 weights for WV or the probabilities for PV). Results are shown in table IV, together with the results of the best single classifier.

The second step consisted of using the confidence values to improve the previously tested combination schemes. The first combination (Maximum Confidence), similar to MV, selects the prediction of the model with higher confidence value. The second one (Weighted Voting Confidence) is similar to WV, but the weights are directly the confidence values of every prediction instead of the F_1 values. The third one (Pairwise Voting Confidence) is a modification of the pairwise voting

TABLE IV

RESULTS OF SEVERAL COMBINING METHODS ON THE TEST SET

Method	Precision	Recall	F_1
bpw-tnh-lin (best single)	91.91%	88.09%	89.96
Majority Voting	93.32%	87.69%	90.42
Weighted Voting	93.36%	87.76%	90.48
Pairwise Voting	93.19%	88.80%	90.94
Maximum Confidence	90.63%	91.28%	90.95
Weighted Voting Confidence	91.22%	90.96%	91.09
Pairwise Voting Confidence	91.75%	91.03%	91.39

scheme, where every pairwise vote is multiplied by the confidence of the prediction of the respective classifiers. Results are shown in table IV. As can be seen, the proposed confidence measure allows to improve the results in a consistent way.

It is worth noting the differences among the precision and recall values of the resulting combinations. Whereas voting methods not using the confidence values have a great difference between precision and recall (the former is more than 4 points higher than the latter), this effect is clearly not so strong when the confidence values are used. Having a good balance between precision and recall is a desirable property for developing real TC systems. In addition, recall values are greater for confidence combinations. This indicates that voting with a confidence factor helps positive predictions to be more important.

VI. CONCLUSIONS AND FUTURE WORK

Several comparisons among margin–based classifiers have been made in a real–world classification problem from the Information Retrieval domain, namely Text Categorization. We observed that the performance of a model seems to be independent of their similarities to other model with similar performance. One surprising result was the observation that the similarities between AdaBoost and SVM classifiers, two margin maximization algorithms, were quite low.

As a consequence of the comparison study, we derived a confidence measure for the prediction of margin–based classifiers. This measure is based on the margin curve. The combination of margin–based classifiers with this confidence measure lead to a marked improvement on the performance of the system, when combined with several well–known combination schemes.

The confidence measures proposed in this work are only a first proposal. This issue deserves further research. In particular, we are interested in defining confidence measures that take into account the percentage of misclassified examples of every classifier. We think that this ideas can lead to better confidence measures, improving the overall performance of a combination scheme.

ACKNOWLEDGMENT

This research has been partially funded by the Spanish Research Department (CICYT's projects: DPI2002-03225, HERMES TIC2000-0335-C03-02, and PETRA TIC2000-1735-C02-02), by the European Commission (MEANING IST-2001-34460), and by the Catalan Research Department (CIRIT's consolidated research group 2001SGR-00254 and research grant 2001FI-00663).

REFERENCES

[1] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press Inc., New York, 1995.

[2] C. C. Chang and C. J. Lin. LIBSVM: A Library for Support Vector Machines. Available at www.csie.ntu.edu.tw/~cjlin/libsvm, 2002.

[3] J. Cohen. "A Coefficient of Agreement for Nominal Scales". *Journal of Educational and Psychological Measurement*, 20, 37–46, 1960.

[4] N. Cristianini and J. Shawe–Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, UK, 2000.

[5] F. Debole and F. Sebastiani. "Supervised term weighting for automated text categorization". *SAC–03, 18th ACM Symposium on Applied Computing*, 784–788, Melbourne, US, 2003.

[6] R. López de Mántaras. "A Distance–Based Attribute Selection Measure for Decision Tree Induction". *Machine Learning*, 6 (1), 81–92, 1991.

[7] H. T. Ng, C. Y. Lim, and S. K. Foo. "A Case Study on Inter–annotator Agreement for Word Sense Disambiguation". *ACL SIGLEX Workshop: Standardizing Lexical Resources*, College Park, MD, USA, 1999.

[8] G. Rätsch. *Robust Boosting via Convex Optimization*. PhD thesis, University of Potsdam, Department of Computer Science, Potsdam, Germany, 2001.

[9] E. Romero and R. Alquézar. "Maximizing the Margin with Feed–forward Neural Networks". *International Joint Conference on Neural Networks*, vol. 1, 743–748, 2002.

[10] E. Romero, L. Màrquez and X. Carreras. "Margin Maximization with Feed–forward Neural Networks: A Comparative Study with SVM and AdaBoost". *Neurocomputing*, 57, 313–344, 2004.

[11] D. E. Rumelhart, G. E. Hinton and R. J. Williams. "Learning Internal Representations by Error Propagation". In Rumelhart, D. E. and McClelland, J. L. (Ed.), *Parallel Distributed Processing* vol. 1, 318–362, MIT Press, 1986.

[12] R. E. Schapire "The Boosting Approach To Machine Learning: An Overview" *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[13] R. E. Schapire and Y. Singer. "Improved Boosting Algorithms Using Confidence–rated Predictions". *Machine Learning* 37 (3), 297–336, 1999.

[14] F. Sebastiani. "Machine Learning in Automated Text Categorization". *ACM Computing Surveys*, 2002.

[15] Vapnik, V.N. *Statistical Learning Theory*. John Wiley & Sons, NY, 1998.