Boosting Parallel Perceptrons for Label Noise Reduction in Classification Problems

Iván Cantador and José R. Dorronsoro [*]

Dpto. de Ingeniería Informática and Instituto de Ingeniería del Conocimiento
Universidad Autónoma de Madrid, 28049 Madrid, Spain

Abstract. Boosting combines an ensemble of weak learners to construct a new weighted classifier that is often more accurate than any of its components. The construction of such learners, whose training sets depend on the performance of the previous members of the ensemble, is carried out by successively focusing on those patterns harder to classify. This fact deteriorates boosting's results when dealing with malicious noise as, for instance, mislabeled training examples. In order to detect and avoid those noisy examples during the learning process, we propose the use of Parallel Perceptrons. Among other things, these novel machines allow to naturally define margins for hidden unit activations. We shall use these margins to detect which patterns may have an incorrect label and also which are safe, in the sense of being well represented in the training sample by many other similar patterns. As candidates for being noisy examples we shall reduce the weights of the former ones, and as a support for the overall detection procedure we shall augment the weights of the latter ones.

1 Introduction

The key idea of boosting methods is to iteratively construct a set $\{h_t\}$ of weak learners that are progressively focused on the "most difficult" patterns of a training sample in order to finally combine them in a final averaged hypothesis $H(X) = \sum_t \alpha_t h_t(X)$. More precisely [7], a starting uniform distribution $D_0 = \{d_0(X)\}$ is progressively updated (see section 3 for more details) to

$$d_{t+1}(X) = \frac{1}{Z_t} d_t(X) e^{-\alpha_t y_X h_t(X)}, \tag{1}$$

where Z_t is a probability normalization constant, $y_X = \pm 1$ is the class label associated to X and the averaging constant $\alpha_t > 0$ is related to the training error ϵ_t of h_t. If a pattern X is incorrectly classified after iteration t, we have $y_X h_t(X) < 0$ and, therefore, boosting iteratively focuses on the incorrectly classified patterns.

When label noisy dataset classification problems (i.e., problems where some pattern labels are incorrect) are considered, this property has two consequences. On the one hand, while boosting has been used with great success in several

[*] With partial support of Spain's CICyT, TIC 01–572, TIN 2004–07676.

applications and over various data sets [2], it has also been shown [3, 4] that it may not yield such good results when applied to noisy datasets. In fact, assume that a given pattern has label noise, that is, although clearly being a member of one class, its label corresponds to the alternate class. Such a label noisy pattern is likely to be repeatedly misclassified by the successive hypotheses which, in turn, will increase its sampling probability, causing boosting to hopelessly concentrate on it and progressively deteriorate the final hypothesis.

Furthermore, mislabeled training examples will tend to have high probabilities as the boosting process advances, which implies that after few iterations, most of the training examples with high weights should correspond to mislabeled patterns, and gives a good motivation to use boosting as a noise filter, as done for instance in [8], where it is proposed that after a number N of rounds, the examples with the highest weights are removed from the training sample. This should allow a more efficient subsequent learning, but also has problems of its own as, for instance, the definition of the exact percentage of presumably noisy examples to be filtered (or, more generally, how to choose a "high enough" weight), or the appropriate choosing of the number N of boosting rounds.

On the other hand, a second issue to be considered on any boosting strategy is the fact that the probabilities of correctly classified examples are progressively diminished. Intuitively, this is a good idea, because the examples that are incorrectly predicted by previous classifiers are chosen more often than examples that were correctly predicted, and boosting will attempt to produce a new classifier more able to handle correctly those patterns for which the current ensemble performance is poor. However, in the presence of medium to high levels of noisy examples, the weak learners may have many difficulties to obtain good separation frontiers, as there may be not enough correctly classified examples in the training samples to do so. In particular, they may be not be able to distinguish between the true mislabeled examples and the incorrectly classified (but well labeled) ones. In other words, it may be sensible to keep an adequate representation of correctly classified patterns in boosting's training sets, as they should make noisy examples to be more easily found. Thus, a "reverse boosting" strategy of keeping well classified patterns, i.e., the "safe" ones, while diminishing label noisy ones may allow a reasonable learning procedure in the label noisy setting. The problem, of course, is how to detect good patterns (even if they may be somewhat redundant) and, more importantly, how to detect noisy patterns. If done properly, an adequate weighting of boosting's exponent should dismiss the latter patterns while keeping the former.

We shall follow this general approach in this work, using Parallel Perceptrons (PPs; [1]) or, more precisely, their activation margins, to detect simultaneously good and noisy patterns. As we shall see in section 2, PPs, a variant of the classical committee machines, not only learn "best" perceptrons but also stabilize their outputs by also learning optimal hidden unit activation margins. These margins shall be used in section 3 to classify training patterns in the just mentioned safe and noisy categories and a third one, borderline patterns, somehow in between of the other ones. This, in turn, can be used to adjust boosting's probability updates. We will do so here by changing the exponent in (1) to $\alpha_t R(X) y_X h_t(X)$, where the $R(X)$ factor will be -1 for safe patterns (increasing their probability) and noisy ones (decreasing now it), and 0 for the bordeline ones (leaving their probability essentially unchanged). The resulting approach will be favorably compared in section 4 with other options such as boosting and bagging of PP and also of standard multilayer perceptrons (MLPs). Finally, the paper will close with a brief summary section and a discussion of further work.

2 Parallel Perceptron Training

Parallel Perceptrons (PP) have the same structure of the well-known committee machines [6]. They are made up of an odd number of standard perceptrons P_i with ± 1 outputs and they have a single one–dimensional output that is simply the sign of the sum of these perceptrons' outputs (that is, the sign of the overall perceptron vote count). They are thus well suited for 2–class discrimination problems, but it is shown in [1] that they can also be used in regression problems. In more detail, assume we are working with D–dimensional patterns $X = (x_1, \ldots, x_D)^t$, where the D–th entry has a fixed 1 value to include bias effects. If the committee machine (CM) has H perceptrons, each with a weight vector W_i, for a given input X, the output of perceptron i is then $P_i(X) = s(W_h \cdot X) = s(act_i(X))$, where $s(\cdot)$ denotes the sign function and $act_i(X) = W_i \cdot X$ is the activation of perceptron i due to X. We then have

$$\sum_{i=1}^{H} P_i(X) = N_+(X) - N_-(X) = \mathcal{N}(X),$$

with $N_\pm(X)$ denoting the number of positive/negative perceptron outputs. The final output $h(X)$ of the CM is $h(X) = s\left(\mathcal{N}(X)\right)$ where we take H to be odd to avoid ties.

We will assume that each input X has an associated ± 1 label y_X and take the output $h(X)$ as correct if $y_X h(X) > 0$. If this is not the case, i.e. whenever $y_X h(X) = -1$, classical CM training ([6], ch. 6) tries to change the smallest number of perceptron outputs so that X could then be correctly classified, choosing those wrong perceptrons for which $|act_i(X)|$ is smallest, and changes their weights by the well-known Rosenblatt's rule:

$$W_i := W_i + \eta y_X X \tag{2}$$

In parallel perceptron training, however, the update (2) is applied to all wrong perceptrons, i.e. those P_i verifying $y_X P_i(X) = -1$.

On the other hand, when a pattern X is correctly classified ($y_X h(X) = 1$), PP training also applies a margin–based output stabilization procedure to those perceptrons for which $0 < y_X act_i(X) < \gamma$. Notice that for them a small perturbation could cause a wrong class assignment and it is convenient to keep their activations $act_i(X)$ away from zero, trying to preserve a margin γ around zero clear from any activation:

$$W_i := W_i + \eta \begin{cases} +X, & \text{if } 0 < act_i(X) < \gamma \\ -X, & \text{if } -\gamma < act_i(X) < 0 \end{cases} \tag{3}$$

The value of the margin γ is dynamically adjusted from a starting value. More precisely, as proposed in [1], after a pattern X is processed correctly, γ is increased to $\gamma + 0.25\eta$ if for all correct perceptrons we have $y_X act_i(X) > \gamma$, while we decrease γ to $\gamma - 0.75\eta$ if $0 < y_X act_i(X) < \gamma$ for at least one correct perceptron.

The training process can be performed either on line or in batch mode. Since we will use PPs in a boosting framework, we shall use the second procedure. Moreover, notice that for the margin to be meaningful, weights have to be normalized somehow; we will make its euclidean norm to be 1 after each batch pass.

In spite of their very simple structure, PPs do have a universal approximation property. As shown in [1], PPs provide results in classification and regression problems quite close to those offered by C4.5 decision trees and only slightly weaker that those of standard multilayer perceptrons (MLPs). Another advantage of using PPs is their very fast training, something quite useful in boosting, where repeated batch trainings will have to be performed.

3 Label Noise Reduction through Parallel Perceptron Boosting

We first discuss how PP's activation margins can be used to detect safe, label noisy and borderline patterns. More precisely, as just described, PPs adaptively adjust these margins, making them to converge to a final value γ that ensures stable PP outputs. Thus, if for a pattern X its i–th perceptron activation verifies $|act_i(X)| > \gamma$, $s(act_i(X))$ is likely to remain unchanged after small perturbations of X. Given the voting outputs of PP, we will accordingly take a pattern as safe if for $\lfloor H/2 \rfloor$ perceptrons P_i (i.e., their majority) we have $y_X act_i(X) > \gamma$, as such an X is likely to be also correctly classified later on. Similarly, if for $\lfloor H/2 \rfloor$ perceptrons we have $y_X act_i(X) < -\gamma$, X is likely to remain wrongly classified, and we will take it to be label noisy. As borderline patterns we will simply take the remaining X. We shall use the notations S_t, N_t and B_t for the safe, noisy and borderline training sets at iteration t. As an example, in figure 1 we show how the safe (squared points) and label noisy (crossed points) patterns of a 2–dimensional XOR problem with 10% of noise level have been detected using a 3–perceptron PP. It shows that almost all label noisy and safe patterns that are quite likely to remain stable in further trainings have been selected.

Let us see how to use this categorization in a boosting–like setting. Recall that boosting's probability updates are given by the rule (1), where $Z_t = \sum_X d_t(X)$,

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right),$$

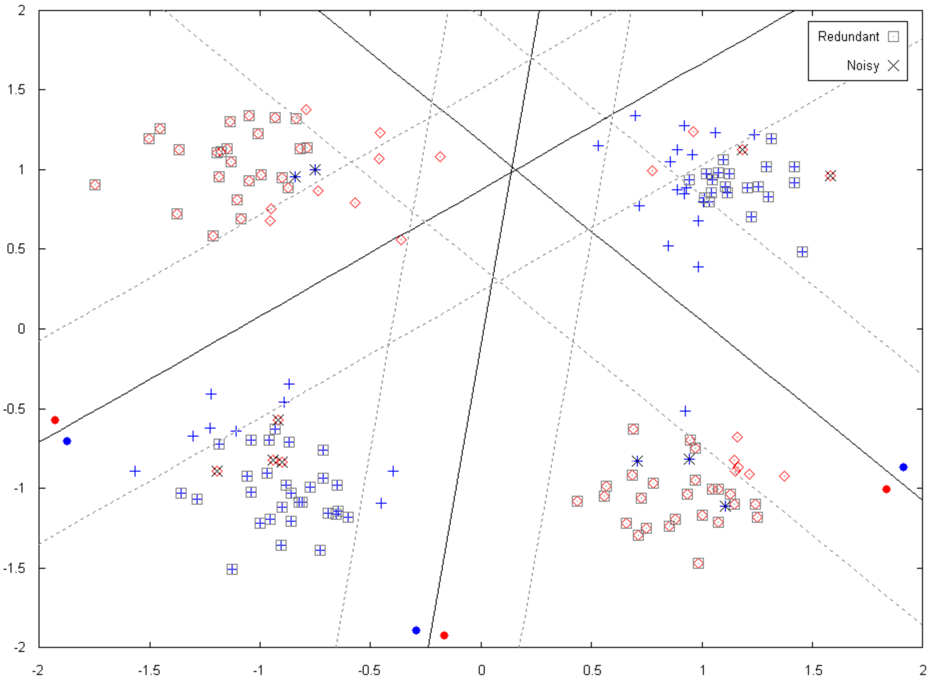

Fig. 1. Safe and mislabeled patterns detection using a PP of 3 standard perceptrons on a 2–dimension XOR problem with 10% of noise level.

and ϵ_t is the iteration error with respect to d_t, i.e.,

$$\epsilon_t = \sum_{\{X \,:\, y_X h_t(X) = -1\}} d_t(X).$$

We introduce safe, noisy and borderline into the boosting process through a pattern dependent factor $R(X)$ in the boosting probability actualization procedure as follows

$$d_{t+1}(X) = \frac{1}{Z'_t} d_t(X) e^{-\alpha_t R(X) y_X h_t(X)}, \tag{4}$$

with Z'_t again a normalization constant. Notice that we recover standard boosting by setting $R(X) = 1$. Now, we want to diminish the influence of label noisy patterns $X \in N_t$, so we put $R(X) = -1$, which should make $d_{t+1}(X) < d_t(X)$, diminishing therefore their importance at the $t+1$ iteration. Moreover, setting also $R(X) = -1$ for safe patterns, we now have $\alpha_t R(X) y_X h_t(X) < 0$, as $y_X h_t(X) > 0$ for them; hence, $d_{t+1}(X) > d_t(X)$, as desired. Finally, for borderline patterns $X \in B_t$ we shall take $R(X) = 0$, that except for changes on the normalization constant, should give $d_{t+1}(X) \simeq d_t(X)$. Notice that except for borderline patterns, the proposed procedure, which we call NR boosting, comes to essentially

Dataset	Noise level	PP bagging	PP boosting	PP NR boosting	MLP bagging	MLP boosting
twonorm	0 %	96.597	96.267	*96.800*	96.733	**97.100**
	5 %	93.600	89.400	**95.933**	*95.700*	93.000
	10 %	90.900	84.600	**95.733**	*94.033*	90.733
	20 %	83.800	75.233	**93.367**	*90.900*	81.967
	30 %	73.800	68.467	**90.167**	*85.200*	72.833
threenorm	0 %	79.367	75.933	78.233	*82.933*	**83.533**
	5 %	77.033	73.700	78.233	**81.367**	*80.100*
	10 %	74.867	71.367	*77.700*	**79.067**	77.300
	20 %	70.100	65.467	**75.000**	*74.733*	71.133
	30 %	66.467	60.967	**71.233**	*69.600*	63.033
ringnorm	0 %	63.900	60.800	64.067	*75.900*	**78.233**
	5 %	61.533	59.100	63.400	**75.400**	*75.300*
	10 %	61.133	59.033	62.100	**72.200**	*71.433*
	20 %	57.967	56.367	60.600	**67.967**	*64.433*
	30 %	56.667	54.000	*59.233*	**61.700**	57.300

Table 1. Accuracies for the PP, MLP bagging and boosting procedures over 3 synthetic datasets with 0%, 5%, 10%, 20% and 30% of noise level in the training samples (best in bold face, second best in italics).

being a "reversed" boosting, as it gives bigger emphasis to correct patterns and smaller to the wrong ones, just the other way around to what boosting does. We shall numerically compare next NR boosting against standard boosting and bagging of PPs and of the much stronger learners given by MLPs.

4 Experiments

In order to have a better control of the noise added, we have used in our experiments three well known synthetically generated datasets of size 300, the twonorm, threenorm and ringnorm datasets, also used in other boosting experiments [2]. We briefly recall their description. They are all 20–dimensional, 2–class problems. In twonorm each class is drawn from a multivariate normal distribution with unit covariance matrix. Class #1 has mean $(a, a, \dots, a)$ and class #2 has mean $(-a, -a, \dots, -a)$ where $a = 2/\sqrt{20}$. In threenorm, class #1 is now drawn from two unit covariance normals, one with mean $(a, a, \dots, a)$ and the other with mean $(-a, -a, \dots, -a)$. Class #2 is drawn from a unit covariance normal with mean $(a, -a, a, -a, \dots, -a)$. Here $a = 2/\sqrt{20}$ too. Finally, in ringnorm, class #1 follows a normal distribution with mean 0 and covariance matrix 4 times the identity and class #2 is a unit covariance normal and mean $(a, a, \dots, a)$ where now $a = 1/\sqrt{20}$. The twonorm and threenorm problems are clearly the easier ones, as they are essentially linearly separable, although there is a greater normal overlapping in threenorm, that gives it a much higher op-

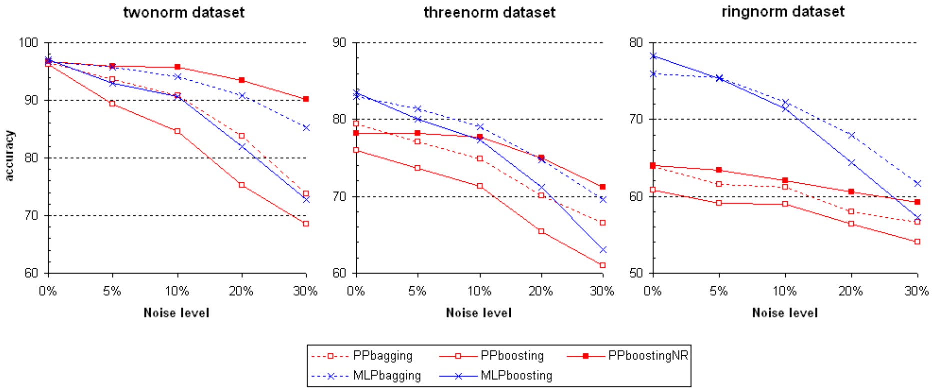

Fig. 2. Graphical comparative results of the accuracies given in table 1. NR boosting accuracies decrease quite slowly; it is thus the more noise robust method.

timal Bayes error probability. Ringnorm is more difficult than the inner; the more concentrated second normal is quite close to the average of the wider first normal. In particular, the Bayes boundary is basically a circle, quite difficult to learn with an hyperplane based method such as PPs (and other simple methods, such as nearest neighbors, linear discriminants or learning vector quantization [5]).

We will compare the results of the PP procedure described in section 3 with those of standard bagging and boosting. These two will also be applied to MLPs. PP training has been carried out as a batch procedure. In all examples we have used 3 perceptrons and parameters $\gamma = 0.05$ and $\eta = 10^{-3}$. As proposed in [1], the η rate does not change if the training error diminishes, but is decreased to 0.9η if it augments. Training epochs have been 250 in all cases; thus the training error evolution has not been taken into account to stop the training procedure. Anyway, this error has an overall decreasing behavior. The MLPs, each with a single hidden layer of 3 units and a learning rate value of $\eta = 10^{-3}$, were trained during 2000 epochs.

In all cases the number of boosting rounds was 10 and we have used 10–times 10–fold cross validation. That is, the overall data set has been randomly split in 10 subsets, 9 of which have been combined to obtain the initial training set, the size of which has. To ensure an appropriate representation of both classes in all the samples, stratified sampling has been used. The final PPs and MLPs' behaviors have been computed on the remaining, unchanged subset, that we keep for testing purposes.

All training samples were artificially corrupted with different levels of classification noise: 5%, 10%, 20% and 30%. Table 1 gives the overall accuracies for the five construction procedures (best values are in bold face, second best in italics). They are also graphically represented in figure 2. In all cases MLPs give best results in absence of noise, with PPs being close seconds for twonorm and threenorm, but not so for the more difficult ringnorm. When noise increases, NR boosting gives best results in twonorm for all noise levels and for the 20% and 30% levels in threenorm; it is second best in the 10% level. It cannot overcome the large head start of MLPs in ringnorm, although is second best for the 30% noise level. On the other hand, NR boosting is clearly the most robust method. For instance, their 30% noise accuracies are just 6.63 points below the noise free ones in twonorm, 7.00 points in threenorm and 4.83 in ringnorm. For MLP bagging (the more robust MLP procedure), its drops are 11.53, 13.39 and 14.20 respectively. This behavior is clearly seen in figure 2.

5 Conclusions and Further Work

We have shown how the concept of activation margin that arises very naturally on PP training can be used to provide a robust approach to label noise reduction. This is done adding an extra factor $R(X)$ to boosting's exponential probability update, with values $R(X) = -1$ for safe and label noisy patterns and $R(X) = 0$ for the borderline ones. The assignment of a patter to each category is done through its activation margins. The resulting procedure has been successfully tested on three well-known synthetic datasets, artificially corrupted with different levels of classification noise. NR boosting has been shown to have a very good overall performance and it is the most robust method, as its accuracies decrease very slowly and it gives the smallest overall drop. Further work will concentrate on the pursuit of a more general approach to malicious noise detection using the ideas of redundant and label noisy patterns categorizations.

References

1. P. Auer, H. Burgsteiner, W. Maass, *Reducing Communication for Distributed Learning in Neural Networks*, Proceedings of ICANN'2002, Lecture Notes in Computer Science 2415 (2002), 123–128.
2. L. Breiman, *Bias, variance and arcing classifiers*, Tech. Report 460, Department of Statistics, University of California, Berkeley, 1996.
3. T. Dietterich, *An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization*, Machine Learning, 40 (2000) 139–158.
4. Y. Freund, R.E. Schapire, *Experiments with a New Boosting Algorithm*, Proceedimgs of the 13th International Conference on Machine Learning (1996), 148–156.
5. D. Meyer, F. Leisch, K. Hornik, *The support vector machine under test*, Neurocomputing, 55 (2003), 169–186.
6. N. Nilsson, **The Mathematical Foundations of Learning Machines**, Morgan Kaufmann, 1990.
7. R.E. Schapire, Y. Freund, P. Bartlett, W.S. Lee, *Boosting the margin: a new explanation for the effectiveness of voting methods*, Annals of Statistics, 26 (1998), 1651–1686.
8. S. Verbaeten, A.V. Assche. *Ensemble methods for noise elimination in classification problems.* In Fourth International Workshop on Multiple Classifier Systems (2003), 317-325.