Computational Intelligence and Bioinspired Systems: 8th International Work-Conference on Artificial Neural Networks, IWANN 2005, Vilanova i la Geltrú, Barcelona, Spain, June 8-10, 2005. Proceedings. Lecture Notes in Computer Science, Volumen 3512. Springer 2005. 208-216.

DOI: http://dx.doi.org/10.1007/11494669_26

Balanced Boosting with Parallel Perceptrons

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 constructs a weighted classifier out of possibly weak learners by successively concentrating on those patterns harder to classify. While giving excellent results in many problems, its performance can deteriorate in the presence of patterns with incorrect labels. In this work we shall use parallel perceptrons (PP), a novel approach to the classical committee machines, to detect whether a pattern's label may not be correct and also whether it is redundant in the sense of being well represented in the training sample by many other similar patterns. Among other things, PP allow to naturally define margins for hidden unit activations, that we shall use to define the above pattern types. This pattern type classification allows a more nuanced approach to boosting. In particular, the procedure we shall propose, balanced boosting, uses it to modify boosting distribution updates. As we shall illustrate numerically, balanced boosting gives very good results on relatively hard classification problems, particularly in some that present a marked imbalance between class sizes.

1 Introduction

As it is well known, boosting constructs a weighted classifier out of possibly weak learners by successively concentrating on those patterns harder to classify. More precisely, it keeps on each iteration a distribution $d_t(X)$ of the underlying X patterns, and after a new hypothesis h_t has been constructed in the t–th iteration, $d_t(X)$ is updated to

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

where $y_X = \pm 1$ is the class label associated to X, Z_t is a probability normalization constant and α_t is related to the training error ϵ_t of h_t (more details in the third section). Therefore, after each iteration boosting concentrates on the patterns harder to classify, as we have $e^{-\alpha_t y_X h_t(X)} > 1$ if $y_X h_t(X) < 0$, i.e., X has been incorrectly classified; as a consequence, the training error ϵ_t will tend to 0 under mild hypothesis on the weak learner [6]. The final hypothesis is the average $h(X) = \sum_t \alpha_t h_t(X)$ of the successively built weak hypotheses h_t.

Boosting has been used with great success in several applications and over various data sets [2]. However, its has also been shown that it may not yield

⋆ With partial support of Spain's CICyT, TIC 01–572

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, would increase its sampling probability and cause boosting to hopelessly concentrate on it. Although this fact may be useful in some instances, its most likely consequence is to deteriorate the final hypothesis. The just described situation may very well happen when dealing with imbalanced data sets, where the number of patterns from one class (that we term the positive one) is much smaller than that from others. There are many examples of this situation, as well as a large literature on this topic, with many techniques having been applied [3, 7]. Most real world classification problems involve imbalanced samples and for them we should expect patterns to fall within three categories: redundant (i.e., easy to classify and likely to be overrepresented in the sample), the just described label noisy and, finally, borderline patterns, i.e., those whose classification could be different after small perturbations and upon which classifier construction should concentrate. To successfully deal with imbalanced data sets it is quite important to detect and handle these three pattern categories correctly.

In this work we introduce a new technique for redundant, label noisy and borderline pattern detection that, in turn, will suggest a new new procedure for boosting's probability update (1) depending on what pattern cathegory X is in. The assignment of X to one of these types is based on another concept of margin that arises naturally in the training of parallel perceptrons (PP), a type of committee machines introduced by Auer et al. in [1] and that will be described in section 2. A key part of the PP training procedure is an output stabilization technique that tries to augment the distance of the activation of a perceptron to its decision hyperplane, i.e., its activation margin, so that small random changes on an input pattern do not cause its being assigned to another class. The activation margins are also learned in some sense during training and can be used for the above classification of training patterns, as it will be described in section 3. In turn, knowing which kind of pattern a given one is 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 reflect the nature of the pattern X. More precisely, $R(X)$ will be 1 for redundant patterns and -1 for noisy ones. We shall consider in section 3 several options for choosing $R(X)$ for borderline patterns; as we shall see in section 4, best results will be obtained by what we shall call balanced boosting, whose results are comparable to those of boosted multilayer perceptrons (MLPs) but with much smaller training times. Finally, the paper will close with a brief summary section and a discussion of further work.

2 Parallel perceptron training

PPs have the same structure of the well known committee machines [5], that is, they are made up of an odd number of standard perceptrons P_i with ±1 outputs, and the machine's one dimensional output 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 $P_i(X) = s(W_i \cdot X) = s(act_i(X))$, where $s(\cdot)$ denotes the sign function and $act_i(X) = W_i \cdot X$ is the activation of perceptron i due to X. The final output $h(X)$ of the CM is $h(X) = s\left(\sum_1^H P_i(X)\right)$ 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 $y_X h(X) = -1$, parallel perceptron training applies the well known Rosenblatt's rule

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

to all wrong perceptrons, i.e. those P_i verifying $y_X P_i(X) = -1$ (η denotes a possibly varying learning rate). Moreover, when a pattern X is correctly classified, 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.

The value of the margin γ is also adjusted dynamically 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. PPs can be trained either on line or in batch mode; since we will use then in a boosting framework, we shall use this second procedure. 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. Moreover, 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). Finally, their training is extremely fast, specially when compared to that of MLPs, something quite useful in boosting, where repeated batch trainings will have to be performed.

3 Boosting parallel perceptrons

As mentioned in the introduction, boosting constructs after each iteration a weak hypothesis h_t over the current distribution d_t, and updates it according to the rule (1), in which $Z_t = \sum_X d_{t+1}(X)$ is a probability normalization, $\alpha_t = \ln\left((1-\epsilon_t)/\epsilon_t\right)/2$, and ϵ_t is the iteration error with respect to d_t, i.e.,

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

Pattern set	neg. boostPP	pos. boostPP	bal. boostPP
R	1	1	1
N	-1	-1	-1
nB^-	1	-1	0
other B	1	1	1

Table 1. The table gives the $R(X)$ labels for training patterns for negative, positive and balanced boosting. All tend to avoid label noisy patterns and their main difference is in the handling of near noisy borderline patters. Standard boosting sets $R(X) = 1$ in all cases.

As mentioned in the introduction, boosting may not yield good results when applied to noisy datasets, as these will be repeatedly misclassified by the successive hypotheses, increasing their sampling probability and causing boosting to hopelessly concentrate on them. in it. On the other hand, PP's activation margins can be used to detect not only label noisy patterns but also those that are redundant and borderline. In more detail, PPs adaptively adjust these margins, making them to converge to a final value γ. 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. Thus if for all i we have $y_X act_i(X) > \gamma$, X is likely to be also correctly classified later on. Those patterns are natural choices to be taken as redundant. Similarly, if for all i we have $y_X act_i(X) < -\gamma$, X is likely to remain wrongly classified, and we will take such patterns as label noisy. The remaining X will be the borderline patterns. We shall use the notations R_t, N_t and B_t for the redundant, noisy and borderline training sets at iteration t. To take into account this categorization, we may introduce a pattern dependent factor $R(X)$ in the boosting probability actualization procedure as follows

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

with Z'_t again a normalization constant. If we set the factor $R(X)$ to be 1, we just recapture standard boosting, while if we want to diminish the influence of label noisy patterns $X \in N_t$, we put $R(X) = -1$; since they are not correctly classified, then $\alpha_t R(X) y_X h_t(X) > 0$ and hence, $d_{t+1}(X) < d_t(X)$. Moreover, we would like to keep boosting focused on borderline patterns, even if they are temporarily misclassified. To do so, we have several options. First we can just proceed as in standard boosting, setting $R(X) = 1$ when $X \in B$; for borderline patterns incorrectly classified this will augment their subsequent probability, while it will diminish it for those well classified. Notice that if the latter are close to the separating hyperplane, they may not be correctly classified afterwards, causing boosting to refocus on them.

However, when dealing with unbalanced datasets, accuracy, that is the percentage of correctly classified patterns, may not be a relevant criterium, as it would be fulfilled by the simple procedure of assigning all patterns to the (possibly much larger) negative class. It may thus be convenient to lessen the impact

		std. boostMLP				std. boostPP			
Dataset	% positives	a	a^+	a^-	g	a	a^+	a^-	g
Ionosphere	35.9	88.40 (0.32)	73.53	96.71	84.3 (1.89)	85.49 (1.51)	68.91	94.85	80.8 (2.02)
Diabetes	34.9	73.80 (0.25)	60.63	80.84	70.0 (1.75)	73.16 (0.97)	57.99	81.42	68.7 (1.06)
Cancer	34.5	95.82 (0.13)	94.19	96.68	95.4 (0.55)	95.64 (0.25)	93.32	96.86	95.1 (0.52)
Vehicle	25.7	83.16 (0.10)	65.35	89.30	76.4 (0.60)	79.04 (1.15)	55.33	87.30	69.5 (1.05)
Glass	13.6	95.82 (0.22)	88.00	97.01	92.4 (1.52)	95.71 (1.06)	86.83	97.13	91.8 (1.84)
Vowel	9.1	99.66 (0.01)	98.22	99.80	99.0 (0.12)	99.46 (0.28)	96.56	99.76	98.1 (0.80)
Thyroid	7.4	98.73 (0.05)	91.30	99.32	95.2 (0.32)	97.33 (0.19)	80.14	98.85	89.0 (1.20)

Table 2. Accuracies and g values for the standard boosting procedures over 7 UCI datasets using MLPs and PPs as learning algorithms. The more complex structure of MLPs gives better accuracies, although those of PPs are quite close in all problems except two.

in training of the more abundant majority negative class. Redundant pattern removal partially takes care of this but it is also interesting to avoid mistraining effects by near noisy label negative patterns, that is, the set nB_t^- of those negative patterns X with a wrong margin $y_X act_i(X) < 0$ in all perceptrons in the t iteration. This can be done by lowering their $d_{t+1}(X)$ probabilities, for which one option is to set $R(X) = -1$; we should expect this to augment the accuracy a^+ of the positive class, while lowering the accuracy a^- of the negative class. We shall call the resulting procedure positive boosting. Of course, we may do the opposite, applying what we may call negative boosting by setting $R(X) = 1$ for $X \in nB_t^-$, which in turn should increase the accuracy a^- of the negative class. A third, more balanced option is to augment the probability of positive borderline patterns (i.e., to set $R(X) = 1$) but to be more "neutral" on the nB_t^- patterns, setting $R(X) = 0$ for them, which will essentially leave their previous probabilities unchanged. While a^+ would then be smaller than in positive boosting, the overall classification should be more balanced. We shall call the resulting procedure balanced boosting. We will measure the balance of positive and negative accuracies using the g coefficient, i.e., the geometric ratio $g = \sqrt{a^+a^-}$ of the positive a^+ and negative a^- accuracies, first proposed in [7]. We shall report next numerical results over seven datasets.

4 Numerical results

We shall use 7 problem sets from the well known UCI database (listed in table 2) referring to the UCI database documentation [4] for more details on these

	std. boostPP			neg. boostPP			pos. boostPP			bal. boostPP		
Dataset	a	a^+	a^-	a	a^+	a^-	a	a^+	a^-	a	a^+	a^-
Ionosphere	**85.49** (1.51)	68.91	94.85	85.09 (1.21)	67.45	95.09	65.97 (1.40)	89.85	52.50	84.97 (1.20)	71.44	92.65
Diabetes	73.16 (0.97)	57.99	81.42	**73.29** (0.88)	59.07	81.04	57.76 (0.96)	94.85	37.56	72.08 (0.88)	73.18	71.49
Cancer	95.64 (0.25)	93.32	96.86	96.03 (0.18)	94.27	97.02	**96.39** (0.22)	99.15	94.91	96.32 (0.16)	96.07	96.50
Vehicle	79.04 (1.15)	55.33	87.30	**79.68** (0.97)	55.04	88.27	72.09 (0.46)	95.52	63.93	78.61 (0.75)	72.41	80.78
Glass	95.71 (1.06)	86.83	97.13	96.09 (0.78)	87.33	97.51	94.76 (0.39)	90.17	95.53	**96.33** (0.88)	89.33	97.46
Vowel	99.46 (0.28)	96.56	99.76	**99.52** (0.35)	97.44	99.73	95.56 (0.13)	99.89	95.12	99.39 (0.24)	98.33	99.50
Thyroid	97.33 (0.19)	80.14	98.85	**97.73** (0.32)	83.42	98.88	94.66 (0.21)	99.56	94.27	97.66 (0.27)	96.60	97.74

Table 3. Accuracies for the boosting procedures over 7 UCI datasets (the lower values give the standard deviations of 10 times 10–fold cross validation); best values in bold face. While it only gives the best result in the glass problem, the overall accuracy of balanced boost is quite close to the best one, while giving a good balance between a^+ and a^-.

problems. Some of them (glass, vowel, vehicle, thyroid) are multi–class problems; to reduce them to 2–class problems, we are taking as the minority classes the class 1 in the vehicle dataset, the class 0 in the vowel data set, and the class 7 in the glass domains (as done in [3]), and merged in a single class both sick thyroid classes. In general they can be considered relatively hard problems and, moreover, some of these problems provide well known examples of highly imbalanced positive and negative patterns, that make difficult classifier construction, as discriminants may tend to favor the (much) larger negative patterns over the less frequently positive ones. This is the case of the glass, vowel, thyroid and, to a lower extent, vehicle problems. In all problems we will take the minority class as the positive one.

PP training has been carried out as a batch procedure. In all examples we have used 3 perceptrons and parameters $\gamma = 0.05$ and $\eta = 10^{-2}$; for the thyroid dataset, we have taken $\eta = 10^{-3}$. 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. We have performed 10 boosting iterations. In all cases 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 positive pattern, stratified sampling has been used. The final

Dataset	std. boostMLP	std. boostPP	neg. boostPP	pos. boostPP	bal. boostPP
Ionosphere	**84.3** (1.89)	80.8 (2.02)	80.1 (1.52)	68.7 (1.85)	81.4 (2.01)
Diabetes	70.0 (1.75)	68.7 (1.06)	69.2 (0.92)	59.7 (0.92)	**72.3** (1.15)
Cancer	95.4 (0.55)	95.1 (0.52)	95.6 (0.49)	**97.0** (0.25)	96.3 (0.38)
Vehicle	76.4 (0.60)	69.5 (1.05)	69.7 (0.99)	**78.1** (0.67)	76.5 (0.88)
Glass	92.4 (1.52)	91.8 (1.84)	92.3 (1.38)	92.8 (1.32)	**93.3** (1.96)
Vowel	**99.0** (0.12)	98.1 (0.80)	98.1 (0.66)	97.5 (0.25)	98.9 (0.48)
Thyroid	95.2 (0.32)	89.0 (1.20)	90.8 (0.50)	96.9 (0.18)	**97.2** (0.35)

Table 4. g values over 7 UCI datasets for the various boosting procedures (the lower values give the standard deviations of 10 times 10–fold cross validation). Best values (in bold face) are given by MLPs and positive boosting in two cases. Balanced boost gives best values in the other three, and it is the second best in the other 4.

PPs' behavior has been computed on the remaining, unchanged subset, that we keep for testing purposes.

As mentioned before, accuracy is a first measure of a classifier's efficiency. Table 3 gives overall, positive and negative accuracies for the four construction procedures (best values are in bold face). It can be seen that negative boosting gives the best results in 4 cases. Standard and balanced boosting give the best accuracy on one problem each, but they are quite close in all others anyway. On the other hand, while positive boosting gives the best accuracy for the cancer dataset, the accuracy it achieves is the lowest in all the other. As it may be expected, it also achieves the highest a^+ values, while standard and, of course, negative boosting strongly favor the negative class. In any case, table 3 also shows that balanced boosting gives the best balance between positive a^+ and negative a^- accuracies. The better inter–class performance of balanced boosting can also be seen in table 4. Balanced boosting achieves the best g values for 5 datasets and is a close second in the other two. Positive boosting gives the highest g for the cancer and vehicle datasets and comes second in two other problems. However, its g performance is quite poor for the diabetes and ionosphere problems. For their part, the g performance of standard and negative boosting comes behind, and while closer for other datasets, it is clearly poorer on the diabetes, vehicle and thyroid cases. The boosting performance of PPs is further compared in tables 2 and 4 with that of standard multilayer perceptrons (MLPs). As they are more powerful, boosted MLPs give clearly better accuracies than boosted PPs in the ionosphere and vehicle problems, worse in the cancer and glass problems and similar in the others. When g values are considered in table 4, balanced boosting gives the best results in 3 problems, while MLP and positive boosting are better over 2 each; in all these 4 cases, balanced boosting g values are second best. In other words, balanced boosting gives the best overall performance among PP boosting methods, and that performance is comparable to that of MLP boosting, that has a much greater complexity and is considerably costlier to train.

5 Conclusions and further work

In this paper we have discussed how the concept of activation margin that arises very naturally on parallel perceptron training can be used to provide a more nuanced approach to boosting. This is done adding an extra factor $R(X)$ to boosting's exponential probability update, whose values depend on the categorization of a given pattern X as redundant, label noisy or borderline obtained in terms of X's activation margins. We set $R(X) = 1$ for redundant and $R(X) = -1$ for label noisy patterns, which causes boosting to lower their subsequent probabilities. Within borderline patterns we consider separately the near label noisy negative patterns. Setting $R(X) = 1$ for them would increase their subsequent probabilities, augmenting thus the negative accuracy a^-, while a^+ would increase if we set $R(X) = 1$. An equilibrium can be obtained setting $R(X) = 0$, thus keeping the probability of X essentially unchanged. The resulting procedure, balanced boosting, gives better PP classifiers in terms of the equilibrium between positive and negative accuracies, while achieving absolute accuracies close to the best achieved by the other methods. This performance is comparable to that of MLP boosting, while PP complexity is lower and training times much shorter than those of MLPs. Further work will concentrate on the effectiveness of general joint PP–boosting approach to malicious noise problems.

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. E. Bauer, R. Kohavi, *An empirical comparison of voting classification algorithms: Bagging, boosting and variants*, Machine Learning 36 (1999), 105–139.
3. M. Kubat, S. Matwin, *Addressing the Curse of Imbalanced Training Sets: One-Sided Selection*, Proceedings of the 14th International Conference on Machine Learning, ICML'97 (pp. 179-186), Nashville, TN, U.S.A.
4. P. Murphy, D. Aha, *UCI Repository of Machine Learning Databases*, Tech. Report, University of Califonia, Irvine, 1994.
5. N. Nilsson, **The Mathematical Foundations of Learning Machines**, Morgan Kaufmann, 1990.
6. 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.
7. J.A. Swets, *Measuring the accuracy of diagnostic systems*, Science 240 (1998), 1285–1293.