in Proc. 2005 IEEE Int. Workshop on Machine Learning for Signal Processing, Mystic, CT

A ROBUST LINEAR PROGRAMMING BASED BOOSTING ALGORITHM

Yijun Sun, Sinisa Todorovic, Jian Li, and Dapeng Oliver Wu

Department of Electrical and Computer Engineering
University of Florida, Gainesville, FL 32611, USA

ABSTRACT

AdaBoost has been successfully used in many signal processing systems for data classification. It has been observed that on highly noisy data AdaBoost leads to overfitting. In this paper, a new regularized boosting algorithm LP$_{\text{norm2}}$-AdaBoost (LPNA), arising from the close connection between AdaBoost and linear programming, is proposed to mitigate the overfitting problem. In the algorithm, the data distribution skewness is controlled during the learning process to prevent outliers from spoiling decision boundaries by introducing a smooth convex penalty function (l_2 norm) into the objective of the minimax problem. A stabilized column generation technique is used to transform the optimization problem into a simple linear programming problem. The effectiveness of the proposed algorithm is demonstrated through experiments on a wide variety of datasets.

1. INTRODUCTION

AdaBoost is a method for improving the accuracy of a learning algorithm (a.k.a. base learner) by calling iteratively the base learner on re-weighted training data, and by combining the so-produced hypothesis functions together to form an ensemble classifier [1, 2]. AdaBoost has been successfully implemented in many signal processing systems [3, 4]. It has been reported that in the low noisy regime, AdaBoost rarely suffers from overfitting problems. However, recent studies with highly noisy patterns have clearly shown that overfitting can occur [5, 6]. In general, there are two distinct cases in which the overfitting phenomena of AdaBoost manifest: (i) when a simple classifier (e.g., C4.5), and (ii) when a powerful classifier (e.g., radial basis functions–RBF) is used as the base learner. In the first case, by investigating the asymptotic behavior of AdaBoost, it has been found that after a large number of iterations, the testing performance may start to deteriorate, despite the continuing increase in the minimum margin of the ensemble classifier [7]. In the second case, AdaBoost quickly leads to overfitting only after a few iterations. These observations indicate that a regularization scheme is needed for AdaBoost in noisy settings.

One method to alleviate the overfitting of AdaBoost is to choose a simple base learner, and implement an early stop strategy. However, in this case, the boosting may not outperform a single well-designed classifier, such as RBF, which undermines the justification for implementing the approach. The second case of overfitting, when a strong base learner is used in AdaBoost, has not been to date well treated in the literature. Only a few algorithms have been proposed to address the problem, among which AdaBoost$_{\text{Reg}}$ achieves the state-of-the-art generalization results on noisy data [6]. Although, in comparison with other regularized algorithms, AdaBoost$_{\text{Reg}}$ empirically shows the best performance [6], it is not known whether it converges, nor what its actual optimization problem is, since the regularization in AdaBoost$_{\text{Reg}}$ is introduced on the algorithm level [2].

In this paper, we present a new regularized boosting algorithm to improve the performance of a strong base learner. Our work is motivated by the close connection between AdaBoost and linear programming (LP) [8]. This connection was used for derivation of LP-AdaBoost [7], where the minimum sample margin is directly maximized. This idea was further explored in [9] by introducing slack variables into an optimization problem in the primal domain. By pursuing a soft margin instead of a hard margin, the resulting algorithm, referred to LP$_{\text{reg}}$-AdaBoost, does not attempt to classify all of the training samples according to their labels (which may be corrupted by noise), but allows for some training errors. In this paper, we provide for a new interpretation of the regularization of LP$_{\text{reg}}$-AdaBoost, which can be viewed as imposing a hard limited penalty function on the data distribution skewness. In contrast, we consider controlling the distribution skewness by using a smooth convex penalty function within the minimax problem formulation. Thus, we propose LP$_{\text{norm2}}$-AdaBoost that has a clear underlying optimization scheme, unlike AdaBoost$_{\text{Reg}}$. Empirical results over a wide range of data demonstrate that our algorithm achieves a similar, and in some cases significantly better classification performance than AdaBoost$_{\text{Reg}}$.

2. REGULARIZED LP BOOSTING ALGORITHMS

Given a set of training data $\mathcal{D}=\{(\mathbf{x}_n, y_n)\}_{n=1}^N \in \mathcal{R}^l \times \{\pm 1\}$, and a class of hypothesis functions $\mathcal{H}=\{h(\mathbf{x}):\mathbf{x}\rightarrow\{\pm 1\}\}$, we are interested in finding an ensemble function $F(\mathbf{x}) =$

$\sum_t \alpha_t h_t(\mathbf{x})$ to classify data into two classes. This can be accomplished by using AdaBoost,[1] which learns combination coefficients and hypothesis functions by performing a functional gradient decent procedure on a cost function of sample margins, $\rho(\mathbf{x}_n) \triangleq y_n F(\mathbf{x}_n)/\sum_t \alpha_t$.

It has been empirically observed that AdaBoost can effectively increase the margin [10]. For this reason, since the invention of AdaBoost, it has been conjectured that AdaBoost, in the limit (i.e., $t \to \infty$), solves the following LP problem:

$$\max_{(\rho,\boldsymbol{\alpha})} \rho, \text{ s.t. } \rho(\mathbf{x}_n) \geq \rho, \ n=1,\cdots,N, \sum_t \alpha_t = 1, \ \boldsymbol{\alpha} \geq 0, \quad (1)$$

referred to as the maximum margin classification scheme. However, recently, the equivalence of the two algorithms has been proven not to hold always [11]. Nevertheless, these two algorithms are closely connected in the sense that both algorithms try to maximize the margin. This observation motivates researchers to design new ensemble classifiers by using some of the well-studied optimization techniques.

Likewise, to derive our regularization scheme, we begin by investigating the minimax problem. For the time being, we assume that the cardinality of $\mathcal{H}$ is finite and is equal to $|\mathcal{H}|$. We define a gain matrix, $\mathbf{Z}$, where $z_{nt} = y_n h_t(\mathbf{x}_n)$. Now, let us look at the following minimax optimization problem:

$$\max_{\boldsymbol{\alpha} \in \Gamma^{|\mathcal{H}|}} \min_{\mathbf{d} \in \Gamma^N} \mathbf{d}^{\mathrm{T}} \mathbf{Z} \boldsymbol{\alpha} \ , \quad (2)$$

where Γ^N is the distribution simplex defined as $\Gamma^N = \{\mathbf{d} : \mathbf{d} \in \mathcal{R}^N, \sum_{n=1}^N d_n = 1, \mathbf{d} \geq 0\}$. The optimization scheme in Eq. (2) can be roughly understood as finding a set of combination coefficients $\boldsymbol{\alpha} \in \Gamma^{|\mathcal{H}|}$, such that the performance of the ensemble classifier in the worst case is optimized. It can be readily shown that this classification scheme leads to the maximum margin classification scheme in Eq. (1). In the separable data case, a large margin is usually conducive to good generalization [12]. However, in the noisy data case with overlapped class distributions and mislabeled training data, the optimization scheme in Eq. (2) can be easily misled by outliers. Consequently, it will produce a classifier with a suboptimal performance. A natural strategy to address this problem is to add a penalty term to the cost function in Eq. (2) to control the data distribution skewness, preventing the algorithm from using all of its resources on learning several hard-to-learn training samples. In the following sections, we present two regularized boosting algorithms that fall within this framework.

2.1. LP$_{\text{reg}}$-AdaBoost (LPRA)

Note that in Eq. (2), the minimization problem is optimized over the entire probability space, which is not sufficiently restrictive. By constraining the distribution into a box, i.e., $\mathbf{d} \leq \mathbf{c}$, we get the following optimization problem:

$$\max_{\boldsymbol{\alpha} \in \Gamma^{|\mathcal{H}|}} \min_{\{\mathbf{d} \in \Gamma^N, \mathbf{d} \leq \mathbf{c}\}} \mathbf{d}^{\mathrm{T}} \mathbf{Z} \boldsymbol{\alpha} \ , \quad (3)$$

where $\mathbf{c}$ is a constant vector. Eq. (3) can be interpreted as finding $\boldsymbol{\alpha}$, such that the classification performance in the worst case within the distribution box is maximized. From Eq. (3), it is straightforward to derive the following primal optimization problem:

$$\max_{\{\rho, \boldsymbol{\lambda}, \boldsymbol{\alpha} \in \Gamma^{|\mathcal{H}|}\}} \rho - \sum_{n=1}^N c_n \lambda_n \ ,$$
$$\text{subject to} \quad \sum_{t=1}^{|\mathcal{H}|} \alpha_t z_{nt} \geq \rho - \lambda_n, \ \lambda_n \geq 0, n=1,\cdots,N. \quad (4)$$

LPRA algorithm [6] is derived from a special case of Eq. (4) by setting $c_1 = \cdots = c_N = C$. The above scheme introduces a nonnegative slack variable, λ_n, into the optimization problem to achieve a sample soft margin, $\rho_s(\mathbf{x}_n) = \rho(\mathbf{x}_n) + \lambda_n$. The relaxation of the hard margin allows the algorithm not to classify all of the training patterns according to their associated labels.

For convenience, we reformulate Eq. (3) as

$$\max_{\boldsymbol{\alpha} \in \Gamma^{|\mathcal{H}|}} \min_{\mathbf{d} \in \Gamma^N} \mathbf{d}^{\mathrm{T}} \mathbf{Z} \boldsymbol{\alpha} + \beta(\|\mathbf{d}\|_\infty), \quad (5)$$

where $\|\cdot\|_p$ is the p-norm and $\beta(P)$ is a function of P defined as: $\beta(P) = 0$ if $P \leq C$, and ∞ if $P > C$. Note that the box defined by $\{\mathbf{d} : \|\mathbf{d}\|_\infty \leq C, \mathbf{d} \in \Gamma^N\}$ is centered at the distribution center $\mathbf{d}_0 = [1/N, \cdots, 1/N]$ (starting point of AdaBoost), and that C controls the skewness of $\mathbf{d}$ between the box boundary and $\mathbf{d}_0$. Eq. (5) indicates that LPRA can be considered as a penalty scheme with a penalty of 0 within the box and ∞ outside the box. Therefore, this scheme is somewhat heuristic, and may be too restrictive. Note that Eq. (5), in fact, provides for a novel interpretation LPRA.

In practical applications, however, the cardinality of $\mathcal{H}$ can be very large or even infinite. Hence, the gain matrix, $\mathbf{Z}$, may not exist in an explicit form and linear programming cannot be implemented directly. This difficulty could be circumvented by using the column generation (CG) technique [9]. It has been shown that by using a commercialized LP package, the CG based LP-Booting algorithm can achieve a comparable performance to that of AdaBoost with respect to both classification quality and solution time [9].

2.2. LP$_{\text{norm2}}$-AdaBoost (LPNA)

Regarding the interpretation of Eq. (5), one plausible strategy to control the skewness of $\mathbf{d}$ is to add a penalty term, $P(\mathbf{d})$, to Eq. (2) as

$$\max_{\boldsymbol{\alpha} \in \Gamma^{|\mathcal{H}|}} \min_{\mathbf{d} \in \Gamma^N} \mathbf{d}^{\mathrm{T}} \mathbf{Z} \boldsymbol{\alpha} + \beta P(\mathbf{d}) \ , \quad (6)$$

where β is a predefined parameter, and $P(\mathbf{d})$ is a function of the distance between query distributions $\mathbf{d}$ and distribution

[1]For a more detailed description of AdaBoost, the interested reader is referred to [2], and references therein.

center $\mathbf{d}_0$. With a mild assumption that $P(\mathbf{d})$ is a convex function of $\mathbf{d}$, it can be shown that Eq. (6) is equivalent to (Generalized Minimax Theorem [13]):

$$\begin{array}{ll} \min_{\{\gamma,\mathbf{d}\in\Gamma^N\}} & \gamma+\beta P(\mathbf{d}) \ , \\ \text{subject to} & \sum_{n=1}^{N} d_n z_{nj} \le \gamma, \ j=1,\cdots,|\mathcal{H}|. \end{array} \tag{7}$$

We refer to Eq. (7) as regularized scheme in the dual domain. In this paper, we define $P(\mathbf{d}) \triangleq \|\mathbf{d}-\mathbf{d}_0\|_2$. Thus, Eq. (7) can be reformulated as

$$\begin{array}{ll} \min_{\{\gamma,\mathbf{d}\in\Gamma^N\}} & \gamma \ , \\ \text{subject to} & \sum_{n=1}^{N} d_n z_{nj} + \beta\|\mathbf{d}-\mathbf{d}_0\|_2 \le \gamma, \\ & j=1,\cdots,|\mathcal{H}|. \end{array} \tag{8}$$

Now, the optimization problem in Eq. (8) can be linearly approximated as follows. First, we define an auxiliary term $s(\mathbf{d}) \triangleq \max_{1\le j\le|\mathcal{H}|} \sum_{n=1}^{N} d_n z_{nj} + \beta\|\mathbf{d}-\mathbf{d}_0\|_2$. Given a set of query distributions $\{\mathbf{d}^{(t)}\}_{t=1}^{T}$, for each query distribution $\mathbf{d}^{(t)}$ there exists a supporting hyperplane, to the epigraph of $s(\mathbf{d})$, given by

$$\gamma = s(\mathbf{d}^{(t)}) + \partial s(\mathbf{d}^{(t)})(\mathbf{d}-\mathbf{d}^{(t)}) \ , \tag{9}$$

where $\partial s(\mathbf{d}^{(t)})$ is the subdifferential of $s(\mathbf{d})$ at $\mathbf{d}^{(t)}$. Due to the convexity of $s(\mathbf{d})$, the supporting hyperplane gives an underestimate of $s(\mathbf{d})$. ¿From Eq. (9), we derive:

$$\begin{aligned} \gamma &= \mathbf{z}_{\cdot t}^{\mathrm{T}}\mathbf{d}^{(t)} + \beta\|\mathbf{d}^{(t)}-\mathbf{d}_0\|_2 + [\mathbf{z}_{\cdot t} + \beta\tfrac{\mathbf{d}^{(t)}-\mathbf{d}_0}{\|\mathbf{d}^{(t)}-\mathbf{d}_0\|_2}]^{\mathrm{T}}(\mathbf{d}-\mathbf{d}^{(t)}), \\ &= [\mathbf{z}_{\cdot t} + \beta\tfrac{\mathbf{d}^{(t)}-\mathbf{d}_0}{\|\mathbf{d}^{(t)}-\mathbf{d}_0\|_2}]^{\mathrm{T}}\mathbf{d} = \tilde{\mathbf{z}}_{\cdot t}^{\mathrm{T}}\mathbf{d} \ , \end{aligned} \tag{10}$$

where $\mathbf{z}_{\cdot t} = [y_1 h_t(\mathbf{x}_1),\cdots,y_N h_t(\mathbf{x}_N)]^{\mathrm{T}}$, and $h_t(\mathbf{x}_n) = \arg\max_{h\in\mathcal{H}} \sum_{n=1}^{N} d_n^{(t)} h(\mathbf{x}_n) y_n$. It follows that Eq. (8) can be linearly approximated as:

$$\begin{array}{ll} \min_{\{\gamma,\mathbf{d}\in\Gamma^N\}} & \gamma \ , \\ \text{subject to} & \tilde{\mathbf{z}}_{\cdot t}^{\mathrm{T}}\mathbf{d} \le \gamma, \quad t=1,\cdots,T \ , \end{array} \tag{11}$$

which is much easier to deal with than the original problem in Eq. (8). The query distributions $\mathbf{d}^{(t)}$ can be obtained by using the column generation technique, as proposed for LP$_{\text{reg}}$-AdaBoost in [9]. Due to the degeneracy of Eq. (11), column generation, however, shows a pattern of slow convergence. The spareness of the optimum solution produces many unnecessary columns, particularly in the initial several iterations. The problem of slow convergence is illustrated in Fig. 1, where, given the columns (constraints) of 1 and 4, the columns of 2 and 3 will not be activated due to the Karush-Kuhn-Tucker (KKT) condition [14]. Consequently, the corresponding hypothesis coefficients α_2 and α_3 are equal to zero. That is, the generation of h_2 and h_3 is not necessary.

One natural idea to alleviate the slow convergence problem is to constrain the solution within a box, centered at the previous solution, also called the BOXSTEP method [15]:

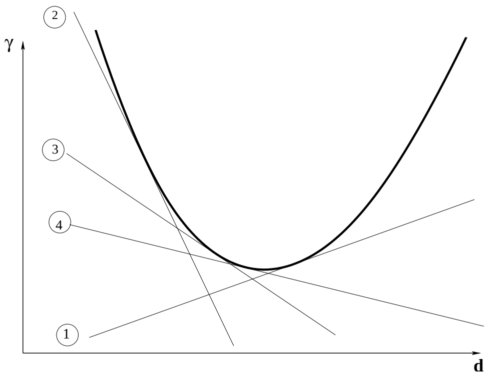

Fig. 1. The slow convergence problem of column generation; the numbers in the circles denote the sequences of generating the columns or constraints.

$$\begin{array}{ll} \min_{\{\gamma,\mathbf{d}\}} & \gamma \ , \\ \text{s.t.} & \tilde{\mathbf{z}}_{\cdot t}^{\mathrm{T}}\mathbf{d} \le \gamma, \ t=1,\cdots,T, \ \mathbf{d}\in\Gamma^N, \ \|\mathbf{d}-\mathbf{d}^{(T)}\|_\infty \le B, \end{array} \tag{12}$$

where parameter B defines the box size. Note that $\|\mathbf{d}-\mathbf{d}^{(T)}\|_\infty \le B \Rightarrow \mathbf{d}^{(T)}-\mathbf{1}B \le \mathbf{d} \le \mathbf{1}B+\mathbf{d}^{(T)}$, which together with $\mathbf{d}\in\Gamma^N$ gives $\max\{\mathbf{d}^{(T)}-\mathbf{1}B,\mathbf{0}\} \le \mathbf{d} \le \mathbf{1}B+\mathbf{d}^{(T)}$. Consequently, the optimization problem in Eq. (12) can be further simplified as the following LP problem:

$$\begin{array}{ll} \min_{\{\gamma,\mathbf{d}\}} & \gamma \ , \\ \text{s.t.} & \tilde{\mathbf{z}}_{\cdot t}^{\mathrm{T}}\mathbf{d} \le \gamma, \ t=1,\cdots,T, \ \sum_{n=1}^{N} d_n = 1, \\ & \max\{\mathbf{d}^{(T)}-\mathbf{1}B,\mathbf{0}\} \le \mathbf{d} \le \mathbf{1}B+\mathbf{d}^{(T)} \ . \end{array} \tag{13}$$

Eq. (13) gives rise to a new LP based boosting algorithm, which we refer to as LP$_{\text{norm2}}$-AdaBoost (LPNA), given in Fig. 2. By using Theorem 4.5.17 and Corollary 4.5.19 in [16], LPNA can be shown to converge to an optimum solution after a finite number of iterations.

The proposed algorithm could be better understood in the primal domain. The dual form of Eq. (11) is given by

$$\begin{array}{ll} \max_{(\rho,\boldsymbol{\alpha}\in\Gamma^T)} & \rho \\ \text{subject to} & \sum_{t=1}^{T}\alpha_t z_{nt} + \beta\sum_{t=1}^{T}\alpha_t\frac{d_n^{(t)}-1/N}{\|\mathbf{d}^{(t)}-\mathbf{d}_0\|_2} \ge \rho, \\ & n=1,\cdots,N \end{array} \tag{14}$$

Similar to Eq. (4), Eq. (14) leads to the following definition of a sample soft margin:

$$\rho_s(\mathbf{x}_n) = \sum_{t=1}^{T}\alpha_t z_{nt} + \beta\sum_{t=1}^{T}\alpha_t\frac{d_n^{(t)}-1/N}{\|\mathbf{d}^{(t)}-\mathbf{d}_0\|_2}, \tag{15}$$

where the term $\left(\beta\sum_{t=1}^{T}\alpha_t\frac{d_n^{(t)}-1/N}{\|\mathbf{d}^{(t)}-\mathbf{d}_0\|_2}\right)$ can be interpreted as "mistrust" in examples. Note that the mistrust is calculated with respect to the initial uniform distribution. This implies that if $d_n^{(t)} \le 1/N$, $t=1,\cdots,T$, then the mistrust can take negative values. As a result, the soft margin provides the mechanism to penalize difficult-to-learn samples with

```
Initialization: $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$; maximum number of iterations $T$; $d_n^{(1)} = 1/N$; parameter $\beta$; box size $B$.

  for $t = 1 : T$
    1. Train base learner with respect to distribution $\mathbf{d}^{(t)}$ and get hypothesis $h_t(\mathbf{x}) : \mathbf{x} \to \{\pm 1\}$.
    2. Solve the optimization problem:
       $(\mathbf{d}^*, \gamma^*) = \arg\min_{\{\gamma, \mathbf{d}\}} \ \gamma$ ,
       subject to $\tilde{\mathbf{z}}_{\cdot j}^{\mathrm{T}} \mathbf{d} \le \gamma, \ j = 1, \cdots, t, \ \sum_{n=1}^N d_n = 1,$
                  $\max\{\mathbf{d}^{(t)} - \mathbf{1}B, \mathbf{0}\} \le \mathbf{d} \le \mathbf{1}B + \mathbf{d}^{(t)}$
    3. Update weights as $\mathbf{d}^{(t+1)} = \mathbf{d}^*$.
  end

Output: $F(\mathbf{x}) = \sum_{t=1}^{\mathrm{T}} \alpha_t^* h_t(\mathbf{x})$, where the $\boldsymbol{\alpha}^*$ are the Lagrangian multipliers from the last LP.
```

Fig. 2. Pseudo-code for LP_{norm2}-AdaBoost algorithm.

large $d_n^{(t)}$ values, and at the same time to award easy-to-learn samples with small $d_n^{(t)}$ values. Since AdaBoost increases the margin of the most hard-to-learn examples at the cost of reducing the margins of the rest of the data [6, 10], by defining the soft margin as in Eq. (15), we seek to reverse the AdaBoost process to some extent, the strength of which is controlled by β.

Interestingly, our soft margin given by Eq. (15), derived in the principled manner from the minimax optimization problem, is very similar to that of $\text{AdaBoost}_{\text{Reg}}$, defined as $\rho_{\text{Reg}}(\mathbf{x}_n) = \sum_{t=1}^T \alpha_t z_{nt} + \beta \sum_{t=1}^T \alpha_t d_n^{(t)}$, which is introduced in AdaBoost on the algorithm level [6]. The main difference is that our soft margin is calculated with respect to the center distribution.

In implementation, if B is chosen too large, LPNA may still slowly converge; if B is too small, the updating of query distributions may not be adequate, affecting the convergence rate of the algorithm. In our experiments, $B \in [\frac{3}{N}, \frac{10}{N}]$ proves appropriate. Throughout, we choose $B = \frac{5}{N}$.

3. EXPERIMENTAL RESULTS

In our experiments, we compare our LPNA with AdaBoost, $\text{AdaBoost}_{\text{Reg}}$, RBF, and LPRA algorithms. For fairness sake, our experimental setup is the same as the one used for evaluation of $\text{AdaBoost}_{\text{Reg}}$ in [6]. We use 12 artificial and real-world data sets originally from the UCI, DELVE and STATLOG benchmark repositions: *banana*, *breast cancer*, *diabetis*, *flare solar*, *german*, *heart*, *image ringnorm*, *splice*, *thyroid*, *titanic*, *twonorm*, and *waveform*. Each data set has 100 realizations of training and testing data. For each realization, a classifier is trained and the test error is computed. The detailed information about the experimental setup and the benchmark data sets can also be found in [17].

The RBF net is used as the base learner. All of the RBF parameters, including the number of the RBF centers and iteration number, are the same as those used in [6]. These parameters, as well as the regularization parameter β are tuned in cross validation. Throughout, the maximum number of iterations of LPNA and LPRA is heuristically set to $T = 150$. The commercialized optimization package XPRESS is used as an LP solver.

We present several examples in which we illustrate the properties of the proposed algorithm. First, we show classification results on *banana* data set, whose samples are characterized by two features. We plot the decision boundaries of AdaBoost and LPNA in the feature space, in Fig. 3. AdaBoost tries to classify each pattern according to its associated label, and forms a "zigzag" decision boundary, illustrating the overfitting phenomenon of AdaBoost. LPNA gives a smooth decision boundary by ignoring some hard-to-learn samples. In the second example, we present the training and testing results, and margin plots of AdaBoost and LPNA on one realization of *waveform* data in Fig. 4. AdaBoost tries to maximize the margin of each pattern, and hence effectively reduces the training error to zero. However, it quickly leads to overfitting. In contrast, LPNA tries to maximize the soft margin, purposefully allowing some hard-to-learn examples to remain with small margins (Note that the hard margin takes a negative value.). Thereby, LPNA effectively alleviates the overfitting problem of AdaBoost.

A more comprehensive comparison of the algorithms is given in Table 1, detailing the average classification results and their standard deviations over the 100 realizations of the 12 datasets. The best results are marked in boldface. From the table, we note the following:

1) AdaBoost performs worse than a single RBF classifier in almost all cases. This is due to the overfitting of AdaBoost. In many cases AdaBoost quickly leads to overfitting only after a few iterations, which clearly indicates that a regularization is needed for AdaBoost.

2) LPNA can significantly improve the performance of RBF, while at the same time avoiding the overfitting problem. In almost all cases, LPNA performs better than AdaBoost.

3) We observe significant improvements in the classification performance of LPNA over $\text{AdaBoost}_{\text{Reg}}$ on some datasets (e.g., *waveform*), whereas for *titanic* and *twonorm*, the results of both algorithms are similar. This indicates the success of our approach. Note that $\text{AdaBoost}_{\text{Reg}}$ has been established as one of the best regularized AdaBoost algorithms, reportedly outperforming Support Vector Machine (with RBF kernel) on the given 12 datasets [6].

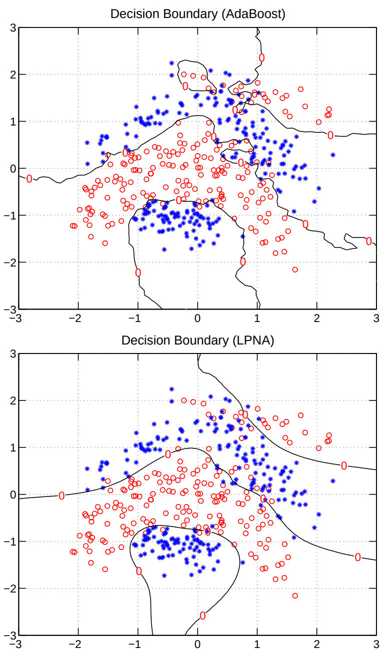

Fig. 3. The decision boundaries of AdaBoost and LPNA on one realization of *banana* data; AdaBoost produces a zigzag decision boundary while LPNA gives smooth decision boundaries.

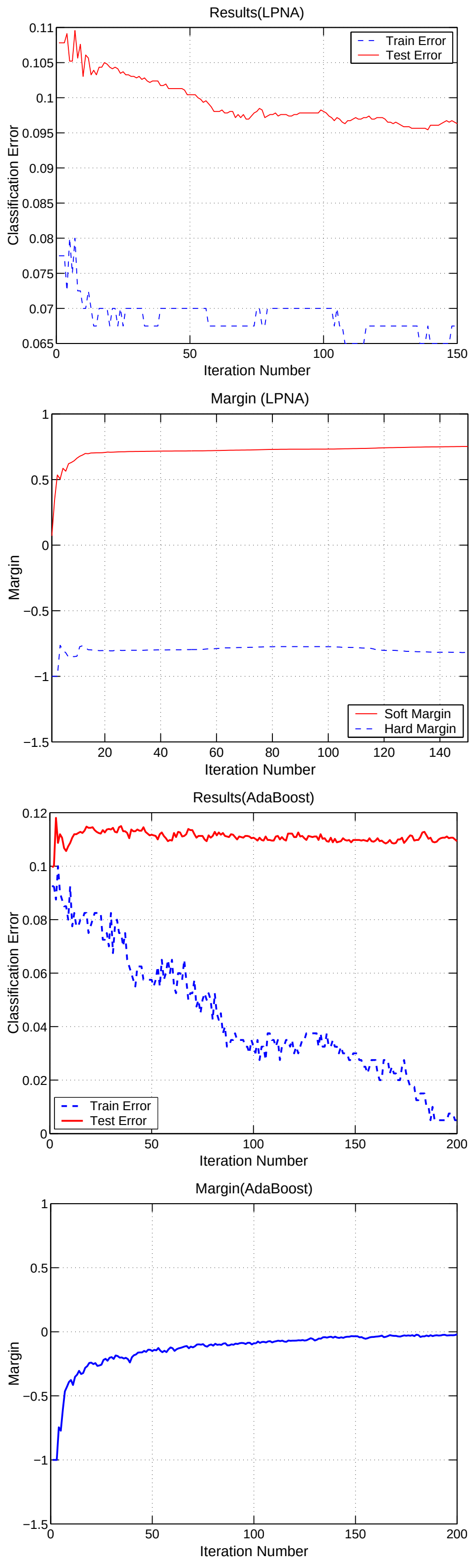

Fig. 4. Training and testing results, and margin plots of AdaBoost and LPNA on one realization of *waveform* data.

4) LPNA is superior to LPRA in almost all cases, while LPRA, in turn, achieves better performance than Ada-Boost. The inferiority of LPRA to LPNA can be explained due to a heuristic hard-limited penalty function used in LPRA.

4. CONCLUSION

In this paper, we have addressed the problem of overfitting in AdaBoost in noisy settings, which may hinder the implementation of AdaBoost for real-world applications. We have proposed a new regularized AdaBoost algorithm – LP_{norm2}–AdaBoost, or short LPNA - by exploring the close connection between AdaBoost and linear programming. The algorithm is based on an intuitive idea of controlling the data distribution skewness in the learning process by introducing a smooth convex penalty function into the objective of the minimax problem. Thereby, outliers are prevented from spoiling decision boundaries in training. We have used the stabilized column generation technique to transform the optimization problem into a simple linear programming problem. Empirical results show that LPNA effectively alleviates the overfitting problem of AdaBoost, and achieves a slightly better overall classification performance than to date the best regularized AdaBoost algorithm called $\text{AdaBoost}_{\text{Reg}}$. Unlike $\text{AdaBoost}_{\text{Reg}}$, where regularization is heuristically introduced on the algorithm level, our algorithm has

Table 1. Classification errors and standard deviations of the RBF, AdaBoost(AB), AdaBoost$_{\text{Reg}}$(AB_R), LP$_{\text{reg}}$-AdaBoost (LPRA), LP$_{\text{norm2}}$-AdaBoost (LPNA). The best resutls are marked in boldface.

	RBF [6]	AB [6]	AB_R [6]	LPNA	LPRA
Banana	10.8±0.6	12.3±0.7	10.9±0.4	**10.7±0.4**	10.9±0.9
Bcancer	27.6±4.7	30.4±4.7	26.5±4.5	**25.9±4.5**	26.7±4.7
Diabetis	24.3±1.9	26.5±2.3	**23.8±1.8**	**23.8±1.8**	24.3±2.0
German	24.7±2.4	27.5±2.5	24.3±2.1	**23.9±2.3**	24.5±2.3
Heart	17.6±3.3	20.3±3.4	**16.5±3.5**	16.9±3.2	17.5±3.6
Ringnorm	1.7±0.2	1.9±0.3	**1.6±0.1**	**1.6±0.2**	1.7±0.2
Fsolar	34.4±2.0	35.7±1.8	**34.2±2.2**	34.3±1.8	34.6±2.0
Thyroid	4.5±2.1	4.4±2.2	4.6±2.2	**4.3±2.2**	4.4±2.1
Titanic	23.3±1.3	22.6±1.2	22.6±1.2	**22.5±1.1**	23.3±0.9
Waveform	10.7±1.1	10.8±0.6	9.8±0.8	**9.4±0.4**	9.8±0.5
Splice	10.0±1.0	10.1±0.5	9.5±0.7	**9.4±0.7**	9.5±0.6
Twonorm	2.9±0.3	3.0±0.3	**2.7±0.2**	**2.7±0.2**	2.8±0.2

a clear underlying optimization scheme.

5. REFERENCES

[1] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[2] R. Meir and G. Rätsch, "An introduction to boosting and leveraging," in *Advanced Lectures on Machine Learning*, S. Mendelson and A. Smola, Eds., pp. 119–184. Springer, 2003.

[3] H. Schwenk, "Using boosting to improve a hybrid HMM/Neural Network speech recognizer," in *Proc. Intl. Conf. Acoustics, Speech, Signal Processing*, Phoenix, AZ, USA, 1999, pp. 1009–1012.

[4] R. Zhang andA. I. Rudnicky, "Improving the performance of an LVCSR system through ensembles of acoustic models," in *Proc. Intl. Conf. Acoustics, Speech, Signal Processing*, Hong Kong, 2003, vol. 1, pp. 876–879.

[5] Thomas G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139–157, 2000.

[6] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft margins for AdaBoost," *Machine Learning*, vol. 42, no. 3, pp. 287–320, 2001.

[7] A. J. Grove and D. Schuurmans, "Boosting in the limit: maximizing the margin of learned ensembles," in *Proc. 15th Nat'l Conf. on Artificial Intelligence*, Madison, WI, USA, 1998, pp. 692–699.

[8] Y. Freund and R. E. Schapire, "Game theory, online prediction and boostin," in *Proc. 9th Annual Conf. Computational Learning Theory*, Desenzano del Garda, Italy, 1996, pp. 325–332.

[9] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor, "Linear programming boosting via column generation," *Machine Learning*, vol. 46, pp. 225–254, 2002.

[10] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[11] C. Rudin, I. Daubechies, and R. E. Schapire, "The dynamics of AdaBoost: Cyclic behavior and convergence of margins," *J. Machine Learning Research*, vol. 5, pp. 1557–1595, Dec 2004.

[12] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, NY, 1998.

[13] I. Ekeland and R. Temam, *Convex Analysis and Variational Problems*, North-Holland Pub. Co., Amsterdam, Holland, 1976.

[14] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, John Wiley and Sons Inc., 2nd edition, 2001.

[15] R. E. Marsten, W. W. Hogan, and J. W. Blankenship, "The BOXSTEP method for large-scale optimization," *Operations Research*, vol. 23, pp. 389–405, 1975.

[16] P. Neame, *Nonsmooth dual methods in integer programming*, Ph.D. thesis, University of Melbourne, Australia, 1999.

[17] G. Raetsch, "IDA benchmark repository," World Wide Web, `http://ida.first.fhg.de/projects/bench/benchmarks.htm`, 2001.