Entropy Regularized LPBoost

Manfred K. Warmuth[1], Karen A. Glocer[1], and S.V.N. Vishwanathan[2]

[1] Computer Science Department
University of California, Santa Cruz
CA 95064, U.S.A
{manfred,kag}@cse.ucsc.edu
[2] NICTA, Locked Bag 8001
Canberra ACT 2601, Australia
SVN.Vishwanathan@nicta.com.au

Abstract. In this paper we discuss boosting algorithms that maximize the soft margin of the produced linear combination of base hypotheses. LPBoost is the most straightforward boosting algorithm for doing this. It maximizes the soft margin by solving a linear programming problem. While it performs well on natural data, there are cases where the number of iterations is linear in the number of examples instead of logarithmic.

By simply adding a relative entropy regularization to the linear objective of LPBoost, we arrive at the Entropy Regularized LPBoost algorithm for which we prove a logarithmic iteration bound. A previous algorithm, called SoftBoost, has the same iteration bound, but the generalization error of this algorithm often decreases slowly in early iterations. Entropy Regularized LPBoost does not suffer from this problem and has a simpler, more natural motivation.

1 Introduction

In the boosting by sampling setting, we are given a training set of $\pm$ labeled examples and an oracle for producing base hypotheses that are typically not very good. A boosting algorithm is able to find a linear combination of hypotheses that is a better classifier than the individual base hypotheses. To achieve this, the boosting algorithms maintain a distribution on the examples. At each iteration, this distribution is given to the oracle, which must return a base hypothesis that has a certain weak guarantee w.r.t. the current distribution on the examples. The algorithm then redistributes the weight on the examples so that more weight is put on the harder examples. In the next iteration, the oracle again provides a base hypothesis with the same weak guarantee for the new distribution and incorporates the obtained base hypothesis into the linear combination, and so forth. When the algorithm stops, it outputs a linear combination of the base hypotheses obtained in all iterations.

When the data is linearly separable, then maximizing the margin is a provably effective proxy for low generalization error [20] and in the inseparable case, maximizing the soft margin is a more robust choice. The soft margin can be maximized directly with a linear program, and this is precisely the approach taken

Y. Freund et al. (Eds.): ALT 2008, LNAI 5254, pp. 256–271, 2008.

by a variant LPBoost [9, 16, 3]. Unless otherwise specified, the name LPBoost stands for this variant. While this algorithm has been shown to perform well in practice, no iteration bounds are known for it. As a matter of fact, the number of iterations required by LPBoost can be linear in the number of examples [24]. So what is the key to designing boosting algorithms with good iteration bounds? Clearly, linear programming is not enough.

To answer this question, let us take a step back and observe that in boosting there are two sets of dual weights/variables: the weights on the fixed set of examples and the weights on the set of currently selected hypotheses. In our case, both sets of weights are probability distributions. It has been shown that AdaBoost determines the weights on the hypotheses by minimizing a sum of exponentials [7]. Many boosting algorithms are motivated by modifying this type of objective function for determining the weights of the hypotheses [4]. Alternatively AdaBoost can be seen as minimizing a relative entropy to the current distribution subject to the linear constraint that the edge of the last hypothesis is nonnegative [11, 12]. All boosting algorithms whose iteration bounds are logarithmic in the number of examples are motivated by either optimizing a sum of exponentials in the hypothesis domain or a cost function that involves a relative entropy in the example domain. This line of research culminated in the boosting algorithm SoftBoost [24], which requires $O(\frac{1}{\epsilon^2} \ln \frac{N}{\nu})$ iterations to produce a linear combination of base hypotheses whose soft margin is within ϵ of the maximum minimum soft margin. Here $\nu \in [1, N]$ is the trade-off parameter for the soft margin.[1] More precisely, SoftBoost minimizes the relative entropy to the initial distribution subject to linear constraints on the edges of all the base hypotheses obtained so far. The upper bound on the edges is gradually decreased, and this leads to the main problem with SoftBoost: the generalization error decreases slowly in early iterations.

Once the relative entropy regularization was found to be crucial in the design of boosting algorithms, a natural algorithm emerged: simply add $\frac{1}{\eta}$ times the relative entropy to the initial distribution to the maximum soft edge objective of LPBoost and minimize the resulting sum. We call this algorithm *Entropy Regularized LPBoost*. A number of similar algorithms (such as ν-Arc [16]) were discussed in Gunnar Rätsch's Ph.D. thesis[14], but no iteration bounds were proven for them even though they were shown to have good experimental performance. Most recently, a similar boosting algorithm was considered by [18, 19] based on the Lagrangian dual of the optimization problem that motivates the Entropy Regularized LPBoost algorithm. However the $O(\frac{1}{\epsilon^{>3}} \ln N)$ iteration bound proven for the recent algorithm is rather weak. In this paper, we prove an $O(\frac{1}{\epsilon^2} \ln \frac{N}{\nu})$ iteration bound for Entropy Regularized LPBoost. This bound is similar to the lower bound of $O(\frac{\ln N}{g^2})$ for boosting algorithms [5] for the hard margin case, where g is the minimum guaranteed edge of the weak learner. In related work, the same bound was proved for another algorithm that involves a tradeoff with the relative entropy [21]. This algorithm, however, belongs to

[1] An earlier version of this algorithm called TotalBoost dealt with the separable case and maximized the hard margin [23].

the "corrective" family of boosting algorithms (which includes AdaBoost) that only update the weights on the examples based on the last hypothesis. LPBoost, SoftBoost and the new Entropy Regularized LPBoost are "totally corrective" in the sense that they optimize their weight based on all past hypothesis. While the corrective algorithms are always simpler and faster on an iteration by iteration basis, the totally corrective versions require drastically fewer iterations and, in our experiments, beat the corrective variants based on total computation time. Also the simple heuristic of cycling over the past hypotheses with a corrective algorithm to make it totally corrective is useful for quick prototyping but in our experience this is typically less efficient than direct implementations of the totally corrective algorithms based on standard optimization techniques.

Outline: In Section 2 we discuss the boosting setting and motivate the Entropy Regularized LPBoost algorithm by adding a relative entropy regularizer to a linear program formulated in the weight domain on the examples. We give the algorithm in Section 3 and discuss its stopping criterion. The dual of Entropy Regularized LPBoost's minimization problem in given in Section 4, and our main result, the iteration bound, is covered in Section 5. Finally, Section 6 contains our experimental evaluation of Entropy Regularized LPBoost and its competitors. The paper concludes with an outlook and discussion in Section 7.

2 The Boosting Setting and LPBoost

In the boosting setting, we are given a set of N labeled training examples (x_n, y_n), $n = 1 \dots N$, where the instances x_n are in some domain $\mathcal{X}$ and the labels $y_n \in \pm 1$. Boosting algorithms maintain a distribution $\mathbf{d}$ on the N examples, so $\mathbf{d}$ lies in the N dimensional probability simplex $\mathcal{S}^N$. Intuitively, the examples that are hard to classify are given more weight. In each iteration $t = 1, 2, \dots$ the algorithm gives the current distribution $\mathbf{d}^{t-1}$ to an oracle (a.k.a. the weak learning algorithm), which returns a new base hypothesis $h^t : \mathcal{X} \to [-1, 1]$ from some base hypothesis class $\mathcal{H}$. The hypothesis returned by the oracle comes with a certain guarantee of performance. This guarantee will be discussed in Section 3.

One measure of the performance of a base hypothesis h^t w.r.t. the current distribution $\mathbf{d}^{t-1}$ is its *edge*, which is defined as $\sum_{n=1}^{N} d_n^{t-1} y_n h^t(x_n)$. When the range of h^t is ± 1 instead of the interval [-1,1], then the edge is just an affine transformation of the weighted error ϵ_{h^t} of hypothesis h^t. A hypothesis that predicts perfectly has edge 1 and a hypothesis that always predicts incorrectly has edge -1, while a random hypothesis has edge approximately 0. The higher the edge, the more useful the hypothesis is for classifying the training examples. The edge of a set of hypotheses is defined as the maximum edge of the set.

It is convenient to define an N-dimensional vector $\mathbf{u}^t$ that combines the base hypothesis h^t with the labels y_n of the N examples: $u_n^t := y_n h^t(x_n)$. With this notation, the edge of the hypothesis h^t becomes $\mathbf{u}^t \cdot \mathbf{d}^{t-1}$.

After a hypothesis h^t is received, the algorithm must update its distribution $\mathbf{d}^{t-1}$ on the examples using $\mathbf{u}^t$. The final output of the boosting algorithm is always a convex combination of base hypotheses $f_{\mathbf{w}}(x_n) = \sum_{q=1}^{T} w_q h^q(x_n)$, where

Algorithm 1. Entropy Regularized LPBoost

1. **Input:** $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, accuracy parameter $\epsilon > 0$, regularization parameter $\eta > 0$, and smoothness parameter $\nu \in [1, N]$.
2. **Initialize:** $\mathbf{d}^0$ to the uniform distribution.
3. **Do for** $t = 1, \ldots$
   (a) Send $\mathbf{d}^{t-1}$ to oracle and obtain hypothesis h^t.
   Set $u_n^t = y_n h^t(\mathbf{x}_n)$
   Assume $\mathbf{u}^t \cdot \mathbf{d}^{t-1} \geq g$, where g need not be known.
   (b) Set $\delta^t = \min_{q=1\ldots t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1})$,
   where $P^t(\mathbf{d}) = \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} + \frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0)$.
   (c) **If** $\delta^t \leq \epsilon/2$ **then** set $T = t - 1$ and break.
   (d) **Else** Update the distribution to $[\mathbf{d}^t, \gamma^t] = \underset{\mathbf{d},\gamma}{\operatorname{argmin}}\, \gamma + \frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0)$,
   s.t. $\mathbf{u}^q \cdot \mathbf{d} \leq \gamma$, for $1 \leq q \leq t$, and $d_n \leq 1/\nu$, for $1 \leq n \leq N$, $\sum_n d_n = 1$.
   This can be done using sequential quadratic programming.
4. **Output:** $f_{\mathbf{w}}(\mathbf{x}) = \sum_{q=1}^{T} w_q h^q(\mathbf{x})$, where the coefficients w_q maximize the soft margin over the hypothesis set $\{h^1, \ldots, h^T\}$ using the LP soft margin problem (2).

w_q is the coefficient of the hypothesis h^q added at iteration q. Ideally, all examples should be on the correct side of the hyperplane defined by the linear combination $f_{\mathbf{w}}$. In other words, for all examples (x_n, y_n), we want $y_n f_{\mathbf{w}}(x_n) > 0$. The *hard margin* of an example (x_n, y_n) measures by how much an example is on the right side and is defined as $y_n f_{\mathbf{w}}(x_n)$. The hard margin of a set of examples is taken to be the minimum margin of the set. When the sign of the convex combination is consistent with the labels, the examples are separated by the hyperplane $f_{\mathbf{w}}$, and this margin is positive. Note that edges are linear in the distribution over the examples and margins are linear in the distribution over the current set of hypotheses.

There is a simple linear programming problem that defines a basic boosting algorithm: update to any distribution that minimizes the maximum edge of the t hypotheses seen so far. That is, $\mathbf{d}^t \in \operatorname{argmin}_{\mathbf{d}\in\mathcal{S}^N} \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d}$. The resulting boosting algorithm is the hard margin version of LPBoost ([9]). By linear programming duality, the minimum-maximum edge equals the maximum-minimum margin:

$$\min_{\mathbf{d}\in\mathcal{S}^N} \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} \;=\; \max_{\mathbf{w}\in\mathcal{S}^t} \min_{n=1,2,\ldots,N} \sum_{q=1}^{t} u_n^q\, w_q. \tag{1}$$

In the case when examples are not separable by a linear combination of the base hypotheses, then the hard margins are naturally replaced by the *soft margins*. The term "soft" here refers to a relaxation of the margin constraint. We now allow examples to lie below the margin but penalize them linearly via slack variables ψ_n. The resulting optimization problem (2) is again a linear program,

$$\max_{\mathbf{w}\in\mathcal{S}^t,\, \boldsymbol{\psi}\geq 0} \min_{n=1,2,\ldots,N} \left(\sum_{q=1}^{t} u_n^q\, w_q + \psi_n \right) - \frac{1}{\nu} \sum_{n=1}^{N} \psi_n, \tag{2}$$

where the trade-off parameter ν is chosen in the range $[1..N]$. Adding slack variables in the hypothesis domain (2) gives rise to the capping constraints $\mathbf{d} \leq \frac{1}{\nu}\mathbf{1}$ in the dual example domain (see e.g. [16, 3] for an early discussion of capping):

$$\min_{\mathbf{d}\in\mathcal{S}^N,\, \mathbf{d}\leq\frac{1}{\nu}\mathbf{1}} \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d}. \tag{3}$$

This suggests that boosting algorithms that cap the weight on the examples do well on inseparable data for the same reasons as algorithms that maximize the soft margin.[2] By linear programming duality, the value at iteration $t = 1, 2, \ldots$ of (2) is equal to the value of its dual (3), and we will denote this value by P^t_{LP}. When $t = 0$, then the maximum is over an empty set of hypotheses and is defined as -1 (i.e. $P^0_{\text{LP}} := -1$).

3 The Entropy Regularized LPBoost Algorithm

In the minimization problem that motivates the main algorithm of this paper, *Entropy Regularized LPBoost*, a relative entropy is added to the linear programming problem in the example domain (3). The relative entropy between distributions $\mathbf{d}$ and $\mathbf{d}^0$ is defined as $\Delta(\mathbf{d}, \mathbf{d}^0) = \sum_{n=1}^N d_n \ln \frac{d_n}{d^0_n}$. The factor $1/\eta$ is a trade-off parameter between the relative entropy and the maximum edge. The modified mini-max problem is defined as follows:

$$\min_{\substack{\mathbf{d}\cdot\mathbf{1}=1 \\ \mathbf{d}\leq\frac{1}{\nu}\mathbf{1}}} \underbrace{\max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} + \frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0)}_{:=P^t(\mathbf{d})}. \tag{4}$$

Note that the constraint $\mathbf{d} \geq \mathbf{0}$ was dropped because the relative entropy is not defined for negative d_n, and therefore the constraints $\mathbf{d} \geq \mathbf{0}$ are enforced implicitly. The relative entropy term makes the objective function $P^t(\mathbf{d})$ strictly convex and therefore the min has a unique solution, which we denote as $\mathbf{d}^t$. We also define $P^t(\mathbf{d})$ when $t = 0$. In this case the maximum is over an empty set of hypotheses and is defined as -1 as before. Thus $P^0(\mathbf{d})$ becomes $-1 + \frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0)$, which is minimized at $\mathbf{d}^0$ and therefore $P^0(\mathbf{d}^0) := -1$.

The Entropy Regularized LPBoost algorithm, shown in Algorithm 1 predicts at trial t with this distribution $\mathbf{d}^t$. Note that in the statement of the algorithm we reformulated (4) into an equivalent convex optimization problem. If the regularization term $\frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0)$ is dropped from the optimization problem then this algorithm becomes the original LPBoost, whose solution is not unique and depends on the LP solver that is used.

[2] When $\nu = 1$, then the capping constraints in (3) are vacuous and this problem is equivalent to the l.h.s. of (1). Similarly, when $\nu = 1$, then the derivatives of the objective of the maximization problem (2) w.r.t. each slack variable ψ_n are all at most zero and $\boldsymbol{\psi} = \mathbf{0}$ is an optimal solution for the maximization over $\boldsymbol{\psi}$. This makes the slack variables disappear and (2) becomes the r.h.s. of (1).

In each iteration t, the boosting algorithm sends a distribution $\mathbf{d}^{t-1}$ to the oracle and the oracle returns a hypothesis h^t from the base hypotheses set $\mathcal{H}$. The returned hypothesis satisfies a certain quality assumption. The strongest assumption is that the oracle returns a hypothesis with maximum edge, i.e. h^t lies in $\operatorname{argmax}_{h\in\mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}^{t-1}$. Iteration bounds for this oracle essentially follow from the work of [22] who use bundle methods to optimize functions of the form

$$\min_{\substack{\mathbf{d} \le \frac{1}{\nu}\mathbf{1} \\ \mathbf{d}\cdot\mathbf{1}=1}} \frac{1}{\eta}\Delta(\mathbf{d},\mathbf{d}_0) + \max_{h\in\mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}.$$

They show that $O(\frac{n}{\epsilon})$ iterations suffice to get within $\epsilon/2$ of the optimum value of this optimization problem. However, the algorithm requires the computation of the gradient of $\max_{h\in\mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}$ evaluated at the current distribution $\mathbf{d}^{t-1}$. The hypothesis returned by the above maximum edge oracle is such a gradient. Setting $\eta = \frac{\ln N}{\epsilon}$ and using a version of the below lemma gives the bound.

This strong oracle is also assumed for the recent corrective algorithm of [21]. In contrast, in this paper we follow [17, 23, 24] and only require a lower bound g on the edge of the hypothesis returned by the oracle. Iteration bounds for this weaker oracle are much harder to obtain.

Assumption on the weak learner: We assume that given any distribution $\mathbf{d}^{t-1} \in \mathcal{S}^N$ on the examples, the oracle returns a hypothesis h^t with edge at least g. We call g the *guarantee* of the oracle. In other words, if $\mathbf{u}^t$ is the vector that combines the base hypothesis h^t with the labels of the examples, then $\mathbf{u}^t \cdot \mathbf{d}^{t-1} \ge g$. In a moment we well discuss the range of admissible values of g.

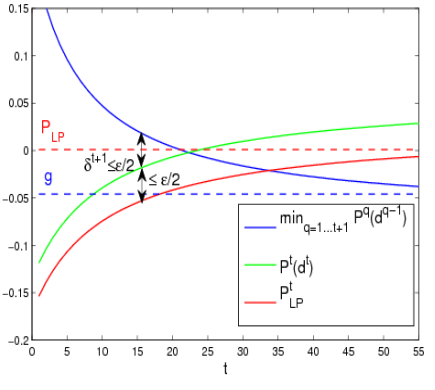

Fig. 1. Depiction of the stopping criterion: $\delta^{T+1} \le \epsilon/2$ implies $g - P^T_{\mathrm{LP}} \le \epsilon$

Stopping criterion: The algorithm monitors $\delta^t := \min_{q=1,2,\ldots,t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1})$, and stops when $\delta^{T+1} \le \epsilon/2$, for a predefined threshold $\epsilon > 0$. Recall that the output of the algorithm is a linear combination $f_{\mathbf{w}}(\mathbf{x}) = \sum_{q=1}^{T} w_q h^q(\mathbf{x})$, where the coefficients w_q are an optimal solution of the LP soft margin problem (3) over the hypothesis set $\{h^1, \ldots, h^T\}$. We first bound the number of iterations T needed before the value P^T_{LP} of this optimization problem gets within ϵ of the guarantee g. All relevant quantities used in the proof are depicted in Figure 1.

Lemma 1. *If $\eta \ge \frac{2}{\epsilon}\ln\frac{N}{\nu}$ in (4), then $\delta^{T+1} \le \epsilon/2$ implies $g - P^T_{LP} \le \epsilon$, where g is the guarantee of the oracle.*

Proof. Since $\Delta(\mathbf{d}, \mathbf{d}^0) \leq \ln \frac{N}{\nu}$ and $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$, we have $\frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0) \leq \epsilon/2$ and

$$P^T(\mathbf{d}^T) \leq P^T_{\text{LP}} + \epsilon/2. \tag{5}$$

On the other hand, from the fact that $\Delta(\mathbf{d}, \mathbf{d}^0) \geq 0$ and the assumption on the weak learner, we know that

$$g \leq \min_{q=1,2,\ldots,T+1} \mathbf{u}^q \cdot \mathbf{d}^{q-1} \leq \min_{q=1,2,\ldots,T+1} P^q(\mathbf{d}^{q-1}).$$

Subtracting $P^T(\mathbf{d}^T)$ and using the stopping criterion we have

$$g - P^T(\mathbf{d}^T) \leq \min_{q=1,2,\ldots,T+1} P^q(\mathbf{d}^{q-1}) - P^T(\mathbf{d}^T) = \delta^{T+1} \leq \epsilon/2.$$

Adding (5) to the above yields $g \leq P^T_{\text{LP}} + \epsilon$. □

Now we must consider the range of the guarantee g that an oracle can achieve. Because $\mathbf{u}^t \in [-1, +1]^N$, it is easy to achieve $g \geq -1$. We claim that the maximum achievable guarantee is $g = P_{\text{LP}}$, where P_{LP} is defined as the value of (3) w.r.t. the entire hypothesis set $\mathcal{H}$ from which oracle can choose. That is,

$$P_{\text{LP}} := \min_{\mathbf{d} \in \mathcal{S}^N} \sup_{\mathbf{d} \leq \frac{1}{\nu}\mathbf{1}} \sup_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d}$$

and therefore, for any distribution $\mathbf{d}$ such that $\mathbf{d} \leq \frac{1}{\nu}\mathbf{1}$, we have $\max_{h \in \mathcal{H}} \mathbf{u}^h \cdot \mathbf{d} \geq P_{\text{LP}}$. Thus, there always exists a hypothesis in $\mathcal{H}$ with edge at least P_{LP}. Also, for any optimal distribution that realizes the value P_{LP}, there is no hypothesis of edge strictly greater than P_{LP}.

For computational reasons, the guarantee g of an oracle may be less that P_{LP}, and therefore we formulate our algorithm and iteration bound for an oracle w.r.t. any guarantee $g \in [-1, P_{\text{LP}}]$. It should be emphasized that our algorithm does not need to know the guarantee g achieved by the oracle.

The line P_{LP} is depicted in Figure 1. The sequence $\langle P^t_{LP} \rangle$ approaches this line from below and the sequence $\langle \min_{q=1\ldots t+1} P^q(\mathbf{d}^{q-1}) \rangle$ approaches the line g from above. When $g < P_{\text{LP}}$, as shown in Figure 1, then both sequences cross, and when $g = P_{\text{LP}}$, then both sequences get arbitrarily close.

Note that the strong oracle discussed earlier which always returns a hypothesis in $\mathcal{H}$ of *maximum edge* w.r.t. the provided distribution clearly has the maximum guarantee P_{LP}. Indeed, on many data sets, this more computationally expensive oracle converges faster than an oracle that only returns a hypothesis of edge P_{LP} (see [17] for an experimental comparison), even though there are no improved iteration bounds known for this stronger oracle.

Our algorithm produces its final linear combination based on the soft margin linear programming problem (3). Alternatively, it could produce a final weight vector $\mathbf{w}$ based on the dual of the regularized minimum-maximum edge problem (4) given in the next section. When $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$, then the regularization term $\frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0)$ is at most $\epsilon/2$, implying that the values of the regularized problems in the example domain are always at most $\epsilon/2$ larger than the corresponding

unregularized problems. Therefore computing the final $\mathbf{w}$ in Entropy Regularized LPBoost via the dual of the regularized problem is also a reasonable choice. In our experiments (not shown) this did not affect the generalization error of the algorithm.

4 The Dual Optimization Problem of Entropy Regularized LPBoost

In this section we compute the Lagrangian dual of (4) and discuss the dual relationship between the the problem of optimizing the distribution on the examples versus optimizing the distribution on the current set of hypotheses. We state the following lemma without proof; the proof is standard and follows from standard arguments (see e.g. [2]).

Lemma 2. *The Lagrangian dual of* (4) *is*

$$\max_{\mathbf{w}} \ \widehat{\Theta}^t(\mathbf{w}, \boldsymbol{\psi}), \quad s.t.\ \mathbf{w} \geq \mathbf{0},\ \mathbf{w} \cdot \mathbf{1} = 1, \boldsymbol{\psi} \geq 0,$$

$$\text{where } \widehat{\Theta}^t(\mathbf{w}, \boldsymbol{\psi}) := -\frac{1}{\eta} \ln \sum_{n=1}^{N} d_n^0 \exp(-\eta(\sum_{q=1}^{t} u_n^q w_q + \psi_n)) - \frac{1}{\nu} \sum_{n=1}^{N} \psi_n.$$

The optimal solution $\mathbf{d}^t$ *of* (4) *can be expressed in terms of the dual variables* $\mathbf{w}^t$ *and* $\boldsymbol{\psi}^t$ *as follows:*

$$d_n^t := \frac{d_n^0 \exp(-\eta(\sum_{q=1}^{t} u_n^q w_q^t + \psi_n^t))}{\sum_{n'} d_{n'}^0 \exp(-\eta(\sum_{q=1}^{t} u_{n'}^q w_q^t + \psi_n^t))}. \tag{6}$$

Furthermore, the value of the primal is equal to the value of the dual. Also, for the optimal primal solution $\mathbf{d}^t$ *and optimal dual solution* $\mathbf{w}^t$*,*

$$\sum_{q=1}^{t} w_q^t \, \mathbf{u}^q \cdot \mathbf{d}^t = \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d}^t.$$

Note that $\widehat{\Theta}^t(\mathbf{w}, \boldsymbol{\psi})$ is a "smoothed" minimum soft margin of the examples for the linear combination $\mathbf{w}$ of the first t hypotheses. As $\eta \to \infty$, this margin becomes the minimum soft margin. Similarly, $P^t(\mathbf{d})$ is the smoothed maximum edge of the first t hypothesis when the distribution on the examples $\mathbf{d}$ lies in the capped probability simplex. Again as $\eta \to \infty$, this edge becomes the maximum edge over the capped simplex. Smoothing is done by different means in the primal versus the dual domain. In the primal it is done by adding one over η times a relative entropy to the max. In the dual, the log partition function smoothes the minimum soft margin.

A smoothing effect can also be seen in the distribution $\mathbf{d}^t$ over the examples. Whereas LPBoost puts its entire weight onto the examples with minimum soft margin w.r.t. the current hypothesis set $\{h^1, \ldots, h^t\}$, Entropy Regularized LPBoost spreads the weight to examples with higher soft margins by taking the

soft min of these margins. The degree of the smoothing is controlled by η. As $\eta \to \infty$, Entropy Regularized LPBoost reverts to an instantiation of LPBoost (i.e. all weight is put on examples with minimum soft margin).

5 Iteration Bound for Entropy Regularized LPBoost

This section contains our main result. For clarity, the number of iterations corresponds to the number of hypotheses incorporated into the final linear combination, rather than calls to the oracle. The number of calls to the oracle is $T+1$ but the number of hypotheses in the final linear combination is T. In other words, the hypothesis h^{T+1} obtained in the last call to the oracle is discarded.

Our first technical lemma bounds the increase $P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1})$ in terms of a quadratic term.

Lemma 3. *If $\eta \geq \frac{1}{2}$, then $P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) \geq \frac{1}{8\eta}(P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}))^2$.*

Proof. First observe that $P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) = \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d}^{t-1} - \max_{q=1,2,\ldots,t-1} \mathbf{u}^q \cdot \mathbf{d}^{t-1}$. Clearly the first max is at least as large as the second. If both are the same, then the lemma trivially holds because $P^t(\mathbf{d}^{t-1}) = P^{t-1}(\mathbf{d}^{t-1})$. If $P^t(\mathbf{d}^{t-1}) > P^{t-1}(\mathbf{d}^{t-1})$, then the first max equals $\mathbf{u}^t \cdot \mathbf{d}^{t-1}$. We can also rewrite the second max by invoking Lemma 2 with $t-1$ instead of t, obtaining

$$P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) = \mathbf{u}^t \cdot \mathbf{d}^{t-1} - \sum_{q=1}^{t-1} w_q^{t-1} \mathbf{u}^q \cdot \mathbf{d}^{t-1} := (\underbrace{\mathbf{u}^t - \sum_{q=1}^{t-1} w_q^{t-1} \mathbf{u}^q}_{:=\mathbf{x}}) \cdot \mathbf{d}^{t-1}.$$

We still need to show that when $\mathbf{x} \cdot \mathbf{d}^{t-1} \geq 0$, $P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) \geq \frac{1}{8\eta}(\mathbf{x} \cdot \mathbf{d}^{t-1})^2$.

By Lemma 2, the value of the optimization problem defining Entropy Regularized LPBoost, $P^t(\mathbf{d})$, equals the value of its dual problem. We begin by lower bounding the increase of this value between successive iterations. Let $\mathbf{w}^t$ and $\boldsymbol{\psi}^t$ denote optimal parameters for the dual problem at iteration t. Because the dual is a maximization problem, $\widehat{\Theta}^t(\mathbf{w}^t, \boldsymbol{\psi}^t) \geq \widehat{\Theta}^t(\mathbf{w}, \boldsymbol{\psi})$ for any suboptimal $\mathbf{w} \in \mathcal{P}^t$ and $\boldsymbol{\psi} \geq \mathbf{0}$. For our lower bound on the value we replace $\boldsymbol{\psi}^t$ by the suboptimal previous value $\boldsymbol{\psi}^{t-1}$ and $\mathbf{w}^t$ by $\mathbf{w}^t(\alpha) = (1-\alpha) \begin{bmatrix} \mathbf{w}^{t-1} \\ 0 \end{bmatrix} + \alpha \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}$, where $\alpha \in [0,1]$:

$$\begin{aligned}
P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) &= \widehat{\Theta}^t(\mathbf{w}^t, \boldsymbol{\psi}^t) - \widehat{\Theta}^{t-1}(\mathbf{w}^{t-1}, \boldsymbol{\psi}^{t-1}) \\
&\geq \widehat{\Theta}^t(\mathbf{w}^t(\alpha), \boldsymbol{\psi}^{t-1}) - \widehat{\Theta}^{t-1}(\mathbf{w}^{t-1}, \boldsymbol{\psi}^{t-1}) \\
&= -\frac{1}{\eta} \ln \sum_{n=1}^{N} d_n^0 \exp\left(-\eta \sum_{q=1}^{t-1} u_n^q w_q^{t-1} - \eta \alpha u_n^t + \eta\, \alpha \sum_{q=1}^{t-1} u_n^q w_q^{t-1} - \eta \psi_n^{t-1} \right) \\
&\quad + \frac{1}{\eta} \ln \sum_{n=1}^{N} d_n^0 \exp\left(-\eta \sum_{q=1}^{t-1} u_n^q w_q^{t-1} - \eta \psi_n^{t-1} \right)
\end{aligned}$$

$$\stackrel{(6)}{=} -\frac{1}{\eta}\ln\sum_{n=1}^{N} d_n^{t-1}\exp\left(-\eta\alpha\underbrace{\left(u_n^t - \sum_{q=1}^{t-1} u_n^q w_q^{t-1}\right)}_{:=x_n}\right).$$

This holds for any $\alpha \in [0,1]$. Since $x_n \in [-2,2]$, then $\exp(x_n) \leq \frac{2+x_n}{4}\exp(-2\eta\alpha) + \frac{2-x_n}{4}\exp(2\eta\alpha)$ and this lets us lower bound the above as

$$-\frac{1}{\eta}\ln\left(\frac{2+\mathbf{x}\cdot\mathbf{d}^{t-1}}{4}\exp\left(-2\eta\alpha\right) + \frac{2-\mathbf{x}\cdot\mathbf{d}^{t-1}}{4}\exp\left(2\eta\alpha\right)\right)$$
$$= 2\alpha - \frac{1}{\eta}\ln\left(1 - \frac{2-\mathbf{x}\cdot\mathbf{d}^{t-1}}{4}(1-\exp(4\eta\alpha)\right)$$

By applying the following inequality from [10]

$$\forall c \in [0,1] \text{ and } r \in \mathbb{R}: \ -\ln(1-c(1-e^r)) \geq -cr - \frac{r^2}{8},$$

the above can be lower bounded by $2\alpha - \frac{2-\mathbf{x}\cdot\mathbf{d}^{t-1}}{4}\,4\alpha - \frac{16\eta\alpha^2}{8}$. This is maximized at $\alpha = \frac{\mathbf{x}\cdot\mathbf{d}^{t-1}}{4\eta}$ which lies in [0,1] because $\mathbf{x}\cdot\mathbf{d}^{t-1} \in [0,2]$ and $\eta \geq \frac{1}{2}$. Plugging this α into the above, we get $\frac{(\mathbf{x}\cdot\mathbf{d}^{t-1})^2}{8\eta}$ as desired. □

Recall the definition of δ^t that we monitor in our proof:

$$\delta^t = \min_{q=1,2,\ldots,t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1}).$$

As done in [22], we now prove a quadratic recurrence for δ^t.

Lemma 4. *If $\eta \geq 1/2$ and $\delta^t \geq 0$, then $\delta^t - \delta^{t+1} \geq \frac{(\delta^t)^2}{8\eta}$, for $1 \leq t \leq T$.*

Proof. We prove this recurrence by bounding the inequality of the previous lemma from above and below. We upper bound the l.h.s. via

$$P^t(\mathbf{d}^t) - P^{t-1}(\mathbf{d}^{t-1}) \leq \underbrace{\min_{1\leq q\leq t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1})}_{\delta^t} - \underbrace{\min_{1\leq q\leq t+1} P^q(\mathbf{d}^{q-1}) - P^t(\mathbf{d}^t)}_{\delta^{t+1}}.$$

To lower bound the r.h.s. of the same inequality, first observe that

$$P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) \geq \min_{1\leq q\leq t} P^q(\mathbf{d}^{q-1}) - P^{t-1}(\mathbf{d}^{t-1}) = \delta^t,$$

Since we assumed $\delta^t \geq 0$, we can lower bound the r.h.s. as

$$\frac{(P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}))^2}{8\eta} \geq \frac{(\delta^t)^2}{8\eta}.$$

□

The lemma requires $\delta^t \geq 0$. The stopping criterion actually assures that $\delta^t > \frac{\epsilon}{2}$, for $1 \leq t \leq T$. Instead of using a recurrence relation, the standard approach would be to show that the value of the underlying optimization problem drops by at least a constant. See e.g. [23, 24] for examples for this type of proof. More precisely, in our case we have

$$P^t(\mathbf{d}^{t-1}) - P^{t-1}(\mathbf{d}^{t-1}) \geq \frac{(\delta^t)^2}{8\eta} \geq \frac{1}{32\eta}\epsilon^2.$$

Unfortunately, for our solution to get ϵ-close to the guarantee g, we need that η is inversely proportional to ϵ and therefore this proof technique only yields the iteration bound of $O(\frac{1}{\epsilon^3} \ln \frac{N}{\nu})$. We shall now see that our recurrence method leads to an improved iterations bound of $O(\frac{1}{\epsilon^2} \ln \frac{N}{\nu})$, which is optimal in the hard margin case (when $\nu = 1$).

Lemma 5. *Let $\langle \delta^1, \delta^2, \ldots \rangle$ be a sequence of non-negative numbers satisfying the following recurrence, for $t \geq 1$: $\delta^t - \delta^{t+1} \geq c(\delta^t)^2$, where $c > 0$ is a positive constant. Then for all integers $t \geq 1$,*

$$\frac{1}{c(t - 1 + \frac{1}{\delta^1 c})} \geq \delta^t.$$

This is Sublemma 5.4 of [1] which is is easily proven by induction.

Theorem 1. *If $\eta = \max(\frac{2}{\epsilon} \ln \frac{N}{\nu}, \frac{1}{2})$ in (4), then Entropy Regularized LPBoost terminates in*

$$T \leq \max(\frac{32}{\epsilon^2} \ln \frac{N}{\nu}, \frac{8}{\epsilon})$$

iterations with a final convex combination of hypotheses for which $g - P^t_{LP} \leq \epsilon$.

Proof. Since $\eta \geq \frac{1}{2}$, we can apply Lemma 4 with $c = \frac{1}{8\eta}$. This give us $\delta^t \leq \frac{8\eta}{t-1+\frac{8\eta}{\delta^1}}$, which can be rearranged to $t \leq \frac{8\eta}{\delta^t} - \frac{8\eta}{\delta^1} + 1 < \frac{8\eta}{\delta^t}$, since $\eta \geq \frac{1}{2}$ and $0 \leq \delta^1 \leq 1$. By the stopping criterion, $\delta^T > \epsilon/2$ and $\delta^{T+1} \leq \epsilon/2$. Thus the above inequality implies that

$$T < \frac{8\eta}{\delta^T} \leq \frac{16\eta}{\epsilon}. \tag{7}$$

By Lemma 1, $\delta^{T+1} \leq \epsilon/2$ implies tolerance ϵ if $\eta \geq \frac{2}{\epsilon} \ln \frac{N}{\nu}$. Plugging $\eta = \max(\frac{2}{\epsilon} \ln \frac{N}{\nu}, \frac{1}{2})$ into the above inequality results in $T \leq \max(\frac{32}{\epsilon^2} \ln \frac{N}{\nu}, \frac{8}{\epsilon})$. Because the aborted iteration $T + 1$ is not counted, we arrive at the iteration bound of the theorem. □

Note that $\frac{2}{\epsilon} \ln \frac{N}{\nu} \geq 1/2$ iff $\epsilon \leq 4 \ln \frac{N}{\nu}$. As pointed out before, when $\eta \to \infty$ then Entropy Regularized LPBoost morphs into a version of LPBoost. Notice that the iteration bound (7) is linear in η. Therefore as $\eta \to \infty$, the iteration bound becomes vacuous.

6 Experimental Evaluation

All of our experiments utilize data from ten benchmark data sets derived from the UCI benchmark repository as previously used in [15]. We compared the soft margin algorithms LPBoost, Entropy Regularized LPBoost, and SoftBoost – with AdaBoost, LogitBoost [8], and BrownBoost[6]. AdaBoost[3] was chosen as a comparator because it is the most common boosting algorithm overall. LogitBoost and BrownBoost were chosen because they are well known and designed for inseparable data.

We used RBF networks as the base learning algorithm.[4] The data comes in 100 predefined splits into training and test sets. For the soft margin algorithms we set $\epsilon = 0.01$. Then for each each of the splits we used 5-fold cross-validation to select the optimal free parameters for each of the algorithms. Finally, in Entropy Regularized LPBoost, η was set to $\frac{2}{\epsilon} \ln \frac{N}{\nu}$. This leads to 100 estimates of the generalization error for each method and data set.

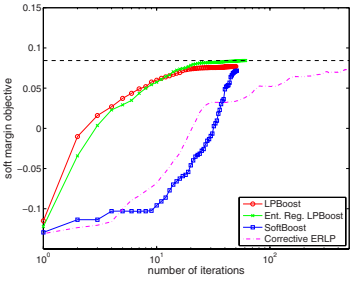

Fig. 2. Soft margin objective vs. the number of iterations on a single run for the Banana data set with $\epsilon = 0.01$ and $\nu/N = 0.1$. In Entropy Regularized LPBoost, $\eta = \frac{2}{\epsilon} \ln \frac{N}{\nu}$. Note that LPBoost is virtually indistinguishable from Entropy Regularized LPBoost, and SoftBoost's margin begins increasing much later than the others. Also, the corrective algorithm converges more slowly than the totally corrective version.

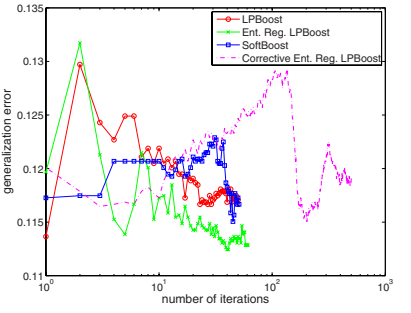

Fig. 3. Generalization error for the same set of algorithms. The corrective algorithm converges more slowly that the totally corrective algorithms. To depict them in the same figure, we use a logarithmic scale of the iteration number in both plots. Note that eventually the generalization error of the corrective algorithm ends up in the same ball-park.

The first step in our empirical analysis of Entropy Regularized LPBoost, is to verify that it actually maximizes the soft margin. Figure 2 shows the soft margin vs. number of iterations for LPBoost, Entropy Regularized LPBoost, and SoftBoost for a single run of the Banana data set from the UCI Benchmark

[3] In AdaBoost, the parameter α_t was chosen to minimize $\sum_n d_n^t \exp(-\alpha u_n^t)$.

[4] The data is from `http://theoval.cmp.uea.ac.uk/~gcc/matlab/index.shtml`. The RBF networks were obtained from the authors of [15], including the hyper-parameter settings for each data set.

Table 1. Generalization error estimates and standard deviations for ten UCI benchmark data sets. As expected, there is no statistically significant difference between LPBoost, SoftBoost, and Entropy Regularized LPBoost, but they outperform AdaBoost and BrownBoost on most data sets.

	AdaBoost	LogitBoost	BrownBoost	LPBoost	SoftBoost	ERLPBoost
Banana	13.3± 0.7	12.4±0.6	12.9±0.7	11.1± 0.6	11.1± 0.5	11.1±0.6
B.Cancer	32.1± 3.8	30.2±4.0	30.2±3.9	27.8± 4.3	28.0± 4.5	28.0±4.4
Diabetes	27.9± 1.5	26.4±1.7	27.2±1.6	24.4± 1.7	24.4± 1.7	24.4±1.7
German	26.9± 1.9	25.3± 1.6	24.8±1.9	24.6± 2.1	24.7± 2.1	24.8±2.2
Heart	20.1± 2.7	19.6±2.8	20.0±2.8	18.4± 3.0	18.2± 2.7	18.3±2.8
Ringnorm	1.9± 0.3*	1.9± 0.2	1.9± 0.2	1.9± 0.2	1.8± 0.2	1.7± 0.2
F.Solar	36.1± 1.5	35.5± 1.5	36.1±1.4	35.7± 1.6	35.5± 1.4	35.5±1.6
Thyroid	4.4± 1.9*	4.7±2.1	4.6± 2.1	4.9± 1.9	4.9± 1.9	4.7±1.9
Titanic	22.8± 1.0	22.8±0.9	22.8±0.8	22.8± 1.0	23.0± 0.8	22.8±1.0
Waveform	10.5± 0.4	10.1±0.4	10.4±0.4	10.1± 0.5	9.8± 0.5	10.1±0.5

repository, with parameters set according to the above description. The results confirm out theoretical intuition that Entropy Regularized LPBoost maximizes the soft margin and more generally, that it does not deviate much from LPBoost. Moreover, it does not start as slowly as SoftBoost. Also included in this plot is the corrective version of Entropy Regularized LPBoost proposed in [21].

Using data from the same run used to generate Figure 2, we also examine generalization error as a function of the number of iterations. The results, shown in Figure 3, confirm that all of the algorithms have similar generalization in the end. Although the corrective algorithm takes many more iterations to converge, its generalization error eventually approaches that of the totally corrective algorithm.

The means and standard deviations for the full benchmark comparison of AdaBoost, LogitBoost, BrownBoost, LPBoost, SoftBoost, and Entropy Regularized LPBoost and are given in Table 1.[5] As we expected, the generalization performances of the soft margin algorithms – LPBoost, Entropy Regularized LPBoost, and SoftBoost – are very similar. In fact, because both SoftBoost and Entropy Regularized LPBoost are intended to approximate LPBoost, we would have been very surprised to see a significant different in their generalization error. The soft margin algorithms outperform AdaBoost on most data sets, while the generalization error of BrownBoost and LogitBoost lie between that of AdaBoost and the soft margin algorithms. Comparing the soft margin algorithms with AdaBoost, LogitBoost and BrownBoost reinforces the idea that maximizing the soft margin results in good algorithms.

Finally, we compared the totally corrective Entropy Regularized LPBoost with the corresponding corrective algorithm of [21]. While a single iteration of the corrective algorithm is faster than a single iteration of the totally corrective algorithm (implemented with sequential quadratic programming), the overall

[5] Note that [15] contains a similar benchmark comparison. It is based on a different model selection setup leading to underestimates of the generalization error. Presumably due to slight differences in the RBF hyper-parameters settings, our results for AdaBoost often deviate by 1-2%.

Table 2. Comparison of total training time in seconds until margin is within $\epsilon = 0.1$ of optimum for LPBoost, Entropy Regularized LPBoost, and the corrective version

	LPBoost	Ent. Reg. LPBoost	Corr. Ent. Reg. LPBoost
Ringnorm	822	2.83e3	2.08e5
Titanic	80	102	2.02e4

running time of the totally corrective algorithm was much faster in our experiments. To demonstrate this, we selected ringnorm and titanic, the largest and smallest datasets of Table 1 respectively, and compared the total running times of LPBoost, Entropy Regularized LPBoost, and Corrective Entropy Regularized LPBoost (Table 2). We used the CPLEX optimizer on a cluster with Intel Xeon X5355 2.66GHz processors. The corrective algorithm of [21] does not come with a stopping criterion, so for a fair comparison we first determined the optimum margin to high precision. We then timed each algorithm until its margin was within $\epsilon = 0.1$ of the optimum. Observe that the corrective algorithm is significantly slower than the totally corrective algorithms on both datasets.

7 Conclusion

We used duality methods instead of Bregman projections to obtain a simple proof for the iteration bound of Entropy Regularized LPBoost[6] We were not able to prove the bound by making fixed progress in each iteration. Instead, following [22] we arrived at a quadratic recurrence equation for the remaining duality gap and, in solving this recurrence, achieved the optimal bound.

We show that the Entropy Regularized LPBoost algorithm has performance comparable to LPBoost and SoftBoost on natural data. It fixes the slow start problem of SoftBoost. We believe that the striking simplicity of the optimization problem for Entropy Regularized LPBoost in which a relative entropy is traded off against a linear objective will lead to further insights into the nature of Boosting and its striking difference to the main other family of learning algorithms (which includes Support Vector Machines (SVMs)) that are motivated by regularizing with the squared Euclidean distance.

Acknowledgments. Thanks to Gunnar Rätsch for valuable cluster time. He also helped us focus on the optimization problem underlying Entropy Regularized LPBoost and develop its dual. The first author was supported by NSF grant CCR 9821087, the second by Los Alamos National Labs and the third by NICTA, Canberra.

[6] This is not too surprising because Bregman projection methods and duality methods are often interchangeable. See [13] for a number of cases where both methods are given for proving iteration bounds on boosting algorithms.

References

[1] Abe, N., Takeuchi, J., Warmuth, M.K.: Polynomial learnability of stochastic rules with respect to the KL-divergence and quadratic distance. IEICE Transactions on Information and Systems E84-D(3), 299–316 (2001)

[2] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)

[3] Demiriz, A., Bennett, K.P., Shawe-Taylor, J.: Linear programming boosting via column generation. Mach. Learn. 46(1-3), 225–254 (2002)

[4] Duffy, N., Helmbold, D.: Potential boosters? In: Solla, S., Leen, T., Müller, K.-R. (eds.) Advances in Neural Information Processing Systems 12, pp. 258–264. MIT Press, Cambridge (2000)

[5] Freund, Y.: Boosting a weak learning algorithm by majority. Inform. Comput. 121(2), 256–285 (1995); In: COLT 1990

[6] Freund, Y.: An adaptive version of the boost by majority algorithm. In: Proceedings of the 12th annual conference on Computational learning theory, pp. 102–113. ACM Press, New York (1999)

[7] Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)

[8] Friedman, J., Hastie, T., Tibshirani, R.: Additive Logistic Regression: a Statistical View of Boosting. The Annals of Statistics 38(2) (2000)

[9] Grove, A.J., Schuurmans, D.: Boosting in the limit: maximizing the margin of learned ensembles. In: AAAI 1998/IAAI 1998, Menlo Park, CA, USA, pp. 692–699 (1998)

[10] Helmbold, D., Schapire, R.E., Singer, Y., Warmuth, M.K.: A comparison of new and old algorithms for a mixture estimation problem. Machine Learning 27(1), 97–119 (1997)

[11] Kivinen, J., Warmuth, M.K.: Boosting as entropy projection. In: Proc. 12th Annu. Conf. on Comput. Learning Theory, pp. 134–144. ACM Press, New York (1999)

[12] Lafferty, J.: Additive models, boosting, and inference for generalized divergences. In: Proceedings of the 12th Annual Conference on Computional Learning Theory, pp. 125–133. ACM Press, New York (1999)

[13] Liao, J.: Totally Corrective Boosting Algorithms that Maximize the Margin. PhD thesis, University of California at Santa Cruz (December 2006)

[14] Rätsch, G.: Robust Boosting via Convex Optimization: Theory and Applications. PhD thesis, University of Potsdam (2001)

[15] Rätsch, G., Onoda, T., Müller, K.-R.: Soft margins for adaboost. Mach. Learn. 42(3), 287–320 (2001)

[16] Rätsch, G., Schölkopf, B., Smola, A., Mika, S., Onoda, T., Müller, K.-R.: Robust ensemble learning. In: Smola, A., Bartlett, P., Schölkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers, pp. 207–219. MIT Press, Cambridge, MA (2000)

[17] Rätsch, G., Warmuth, M.: Efficient margin maximizing with boosting. Journal of Machine Learning Research 6, 2131–2152 (2005)

[18] Rudin, C., Schapire, R., Daubechies, I.: Boosting based on a smooth margin. In: Proceedings of the 17th Annual Conference on Computational Learning Theory, pp. 502–517 (2004)

[19] Rudin, C., Schapire, R., Daubechies, I.: Analysis of boosting algorithms using the smooth margin function. The Annals of Statistics 6(35), 2723–2768 (2007)

[20] Schapire, R., Freund, Y., Bartlett, P., Lee, W.: Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics 26(5), 1651–1686 (1998)
[21] Shalev-Shwartz, S., Singer, Y.: On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms. In: Proceedings of the 21st annual conference on Computational learning theory, pp. 311–321. Omicron (2008)
[22] Smola, A., Vishwanathan, S.V.N., Le, Q.: Bundle methods for machine learning. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20, pp. 1377–1384. MIT Press, Cambridge (2008)
[23] Warmuth, M., Liao, J., Rätsch, G.: Totally corrective boosting algorithms that maximize the margin. In: ICML 2006, pp. 1001–1008. ACM Press, New York (2006)
[24] Warmuth, M.K., Glocer, K., Rätsch, G.: Boosting algorithms for maximizing the soft margin. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20. MIT Press, Cambridge (2007)