Totally Corrective Boosting Algorithms that Maximize the Margin

Manfred K. Warmuth MANFRED@CSE.UCSC.EDU
Jun Liao LIAOJUN@CSE.UCSC.EDU
University of California at Santa Cruz, Santa Cruz, CA 95060, USA

Gunnar Rätsch GUNNAR.RAETSCH@TUEBINGEN.MPG.DE
Friedrich Miescher Laboratory of the Max Planck Society Spemannstr. 35, 72076 Tübingen, Germany

Boosting, Margins, Convergence, Relative Entropy, Bregman Divergences, Bregman Projection

Abstract

We consider boosting algorithms that maintain a distribution over a set of examples. At each iteration a weak hypothesis is received and the distribution is updated. We motivate these updates as minimizing the relative entropy to the last distribution subject to linear constraints. For example AdaBoost constrains the *edge* of the last hypothesis w.r.t. the updated distribution to be at most $\gamma = 0$ (equivalently its weighted error is constrained to be half). In some sense, AdaBoost is "corrective" w.r.t. the last hypothesis. A more principled Boosting method is to be "totally corrective" in that the edges of all the past hypotheses are constrained to be at most γ, where γ is suitably adapted.

Using new techniques, we prove the same iteration bounds for the totally corrective algorithms as for their corrective versions. Moreover with adaptive γ, the algorithms can be shown to provably maximize the margin. Experimentally, the totally corrective versions clearly outperform the corrective ones and are competitive with the most efficient algorithm so far, LPBoost, for which there are no iteration bounds known.

*The first author was partially funded by the NSF grant CCR-9821087. The first two authors were partially funded by UC Discovery grant ITl03-10127 and Telik Inc. grant ITl03-10110. Part of this work was done while the third author was visiting UC Santa Cruz. This author was partially funded by DFG under contract JA 379/91, JA 379/71, MU 987/1-1 and by EU in the NeuroColt II project.

1. Introduction

The goal of this paper is to characterize boosting algorithms by the underlying optimization problems rather than the approximation algorithms that solve these problems. One of the most common boosting algorithms is AdaBoost (Freund & Schapire, 1997; Schapire & Singer, 1999). It can be viewed as minimizing the relative entropy to the last distribution subject to the constraint that the edge of the last hypothesis is zero (equivalently its weighted error is half) (Kivinen & Warmuth, 1999; Lafferty, 1999). One of the important properties of AdaBoost is that it approximately maximizes the margin of the examples and therefore seems to exploit the theoretical generalization bounds that improve with this margin (Schapire et al., 1998). AdaBoost only approximately maximizes the margin (Rudin et al., 2004a). However a similar algorithm called AdaBoost$^*_\nu$ has been developed recently (Rätsch & Warmuth, 2005) that provably maximizes the margin and has analogous iteration bounds as AdaBoost. The algorithm AdaBoost$^*_\nu$ also processes only a single constraint in each iteration: the edge of the last hypothesis must to be at most γ, for an adaptive choice of γ.

A natural idea is to process all t past hypotheses at the end of iteration t, i.e. minimize the relative entropy to the last distribution on the examples subject to the constraints that the edges of all t past hypotheses are at most γ. Such algorithms were proposed by Kivinen and Warmuth (1999) and called "totally corrective". However in that paper only $\gamma = 0$ was considered, which leads to an infeasible optimization problem when the training data is separable. Building on the work of Rätsch and Warmuth (2005), we now adapt γ during the totally corrective algorithm so that the margin is approximately maximized. We call our new algorithm TotalBoost$_\nu$.

AdaBoost$^*_\nu$ can be used as a heuristic for implementing TotalBoost$_\nu$ by doing multiple passes over all past hypotheses before adding a new one. However we can show that this heuristic often requires several millions of iterations and is considerably less efficient than a vanilla *sequential quadratic optimization* approach for solving the optimization problem underlying TotalBoost$_\nu$.

Besides developing new proof methods for iteration bounds of boosting algorithms that parallel the methods for proving relative loss bounds for online algorithms (Kivinen & Warmuth, 1997), this paper also does an experimental comparison of the algorithms. We show that while TotalBoost$_\nu$ has the same iteration bound as AdaBoost$^*_\nu$, it often requires several orders of magnitudes fewer iterations. When there are many similar weak hypotheses, the totally corrective algorithms has an additional advantage: assume we have 100 groups of 100 weak hypotheses each, where the hypotheses within each group are very similar. TotalBoost$_\nu$ picks at most one hypothesis from each group, whereas the algorithms that process one constraint at a time often come back to the same group and choose many more members from the same group. Therefore the number of weak hypotheses used in the output convex combination is much smaller for the totally corrective algorithms, making them better suited for the purpose of feature selection.

Perhaps one of the simplest boosting algorithms is LPBoost: it is totally corrective, but unlike TotalBoost$_\nu$, it uses no entropic regularization. Also, the upper bound γ on the edge is chosen to be as small as possible in each iteration, whereas in TotalBoost$_\nu$ it is decreased more moderately. Experimentally, we have identified cases where TotalBoost$_\nu$ requires considerably fewer iterations than LPBoost, which suggests that either the entropic regularization or the moderate choice of γ is helpful for more than just for proving iteration bounds.

2. Preliminaries

Assume we are given N labeled examples $(\mathbf{x}_n, y_n)_{1\le n\le N}$, where the examples are from some domain and the labels y_n lie in $\{\pm 1\}$. Boosting algorithms maintain a distribution $\mathbf{d}$ on the examples. Intuitively d_n is large if the example is "hard". The standard assumption made on the weak learning algorithm for the PAC analysis of Boosting algorithm is that the weak learner returns a hypothesis h from a fixed set $\mathcal{H}$ whose error ϵ is smaller than $\frac{1}{2}$. Here the error ϵ of a ± 1 valued hypothesis is the total weight on all the examples that are misclassified.

When the range of a hypothesis h is the entire interval $[-1,+1]$, then the *edge* $\gamma_h(\mathbf{d}) = \sum_{n=1}^N d_n y_n h(\mathbf{x}_n)$ is a more convenient quantity for measuring the quality of h. This edge is an affine transformation of the error for the case when h has range ± 1: $\epsilon_h(\mathbf{d}) = \frac{1}{2} - \frac{1}{2}\gamma_h(\mathbf{d})$ and $\epsilon_h(\mathbf{d}) \le \frac{1}{2}$ iff $\gamma_h(\mathbf{d}) \ge 0$. The edge for a set of hypotheses is always the maximum over the hypotheses.

Boosting algorithms produce a convex combination of weak hypotheses: $f_{\boldsymbol{\alpha}}(\mathbf{x}) := \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$, where h_t is the hypothesis added in iteration t and α_t its coefficient. The margin of a given example $(\mathbf{x}_n, y_n)$ is defined as $y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n)$. The margin of a set of examples is always the minimum over the examples. For sake of simplicity of the presentation[1] we assume that $\mathcal{H}$ has finite size M. If we combine all hypotheses from $\mathcal{H}$, then the following classical theorem (von Neumann, 1928) establishes the connection between margins and edges:

Theorem 1 (Min-Max-Theorem)

$$\begin{aligned}\gamma^* &= \min_{\mathbf{d}} \max_{m=1,\dots,M} \sum_{n=1}^N d_n y_n h_m(\mathbf{x}_n) \\ &= \max_{\boldsymbol{\alpha}} \min_{n=1,\dots,N} y_n \sum_{m=1}^M \alpha_m h_m(\mathbf{x}_n) \quad = \rho^*,\end{aligned}$$

where $\mathbf{d} \in \mathcal{P}^N$, $\boldsymbol{\alpha} \in \mathcal{P}^M$ *and* $M = |\mathcal{H}|$. *Here* $\mathcal{P}^k$ *denotes the* k-dimensional probability simplex.

Thus, the minimum edge γ^* that can be achieved over all possible distributions $\mathbf{d}$ of the training set is equal to the maximum margin ρ^* of any convex combination of hypotheses from $\mathcal{H}$. At the end of trial t, the set of previous hypotheses is $\mathcal{H}_t = \{h_1, \dots, h_t\}$. Because of the above duality and the fact that $\mathcal{H}_t \subseteq \mathcal{H}$,

$$\gamma_t^* := \min_{\mathbf{d}} \max_{h\in\mathcal{H}_t} \gamma_h(\mathbf{d}) \quad \le \quad \gamma^* = \rho^* \ . \tag{1}$$

The non-decreasing sequence $\{\gamma_t^*\}_t$ converges to ρ^* from below. Also, for any non-optimal distributions $\mathbf{d}$ and hypothesis weights $\boldsymbol{\alpha}$ we always have

$$\max_{h\in\mathcal{H}} \gamma_h(\mathbf{d}) \ > \ \gamma^* = \rho^* \ > \ \min_{n=1,\dots,N} y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n) =: \rho(\boldsymbol{\alpha}). \tag{2}$$

In view of the above, we assume that the edge γ_t of hypothesis h_t provided by the weak learner at trial t is always at least ρ^*.

One of the most bare-bones boosting algorithms is LPBoost (Algorithm 1) proposed by Grove and Schuurmans (1998); Bennett et al. (2000). It uses linear programming to constrain the edges of the past t weak hypotheses to be at most γ_t^*, which is as small as possible. No iteration bound is known for this algorithm,

[1] For example, the Min-Max-Theorem still holds if $\mathcal{H}$ is compact (see discussion in Rätsch & Warmuth, 2005).

and also the performance can very much depend on the LP solver used (interior vs. boundary solutions).

Our algorithms are motivated by the minimum relative entropy principle of Jaynes: find a solution subject to some linear constraints and among all feasible solutions choose the one that minimizes a relative entropy to the last distribution $\mathbf{d}^t$. The relative entropy is defined as follows: $\Delta(\mathbf{d}, \widetilde{\mathbf{d}}) = \sum_n d_n \ln \frac{d_n}{\widetilde{d}_n}$. There are two totally corrective versions of the algorithm: one that knows the target margin ρ^* and one that does not. The one that does (called TotalBoost$_\nu^{\rho^*}$; Algorithm 3), simply constrains the edges of the previous hypotheses to be at most $\rho^* - \nu$, where ν is a given precision parameter. Our main algorithm, TotalBoost$_\nu$ (Algorithm 2) does not know ρ^*. It maintains the estimates $\widehat{\gamma}_t = \left(\min_{q=1}^t \gamma_q\right) - \nu$ and constrains the edges of the past hypotheses to be at most $\widehat{\gamma}_t$. Note that the sequence $\{\widehat{\gamma}_t\}_t$ is non-increasing and by our assumption $\gamma_t \geq \rho^*$, its elements are at least $\rho^* - \nu$.

The relative entropy is an example of a Bregman divergence. Our iteration bound relies on a generalized Pythagorean theorem for such divergences (Censor & Zenios, 1997; Herbster & Warmuth, 01) and therefore we need to require that the current distribution $\mathbf{d}_t$ (the one we are projecting onto the linear constraint) lies in the interior of domain $\mathcal{P}^N$.

3. Termination Guarantees

When the algorithms break, we need to guarantee that the margin w.r.t. the current hypothesis set is at least $\rho^* - \nu$. Furthermore, for all algorithms except LPBoost we need to assure that $\mathbf{d}^t$ lies in the interior of $\mathcal{P}^N$. Therefore the algorithms also need to break when the updated distribution $\mathbf{d}^{t+1}$ lies at the boundary of $\mathcal{P}^N$.

TotalBoost$_\nu^{\rho^*}$ is given ρ^* and constrains the edges of all past hypotheses to be at most $\rho^* - \nu$. When these become infeasible, the edge γ_t^* w.r.t. the current hypotheses set is larger than $\rho^* - \nu$. The algorithm also breaks when the solution $\mathbf{d}^{t+1}$ of the minimization problem lies at the boundary of $\mathcal{P}_N$ (i.e. the distribution has a zero component). In this case $\gamma_t^* = \rho - \nu$, because if $\gamma_t^* < \rho - \nu$, then all constraints would have slack and the solution $\mathbf{d}$ that minimizes the divergence $\Delta(\mathbf{d}, \mathbf{d}^t)$ ($\mathbf{d}^t$ lies in the interior of $\mathcal{P}^N$) would also lie in the interior. Thus whenever the algorithm breaks, we have $\rho^* - \nu \leq \gamma_t^*$. TotalBoost$_\nu^{\rho^*}$ outputs a convex combination of the hypotheses $\{h_1, \ldots, h_T\}$ that maximizes the margin. By duality, the value of this margin equals the minimum edge γ_t^* and therefore TotalBoost$_\nu^{\rho^*}$ is guaranteed to output a hypothesis of margin larger than $\rho^* - \nu$.

Algorithm 1 LPBoost algorithm

1. **Input:** $S = \langle (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \rangle$, desired accuracy ν
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots,$
   (a) Train classifier on $\{S, \mathbf{d}^t\}$ and obtain hypothesis $h_t : \mathbf{x} \mapsto [-1, 1]$ and let $u_n^t = y_n h_t(\mathbf{x}_n)$
   (b) Calculate the edge γ_t of h_t: $\gamma_t = \mathbf{d}^t \cdot \mathbf{u}^t$,
   (c) Set $\widehat{\gamma}_t = (\min_{q=1,\ldots,t} \gamma_q) - \nu$
   (d) Compute γ_t^* as in (1) and set $\mathbf{d}^{t+1}$ to any distribution $\mathbf{d}$ for which $\mathbf{u}^q \cdot \mathbf{d} \leq \gamma_t^*$, for $1 \leq q \leq t$
   (e) **If** $\gamma_t^* \geq \widehat{\gamma}_t$ **then** $T = t$ and break
4. **Output:** $f_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$, where the coefficients α_t realize margin γ_T^*.

TotalBoost$_\nu$ does not know ρ^*. It breaks if its optimization problem becomes infeasible, which happens when γ_t^* is larger than the estimate $\widehat{\gamma}_t$. This estimate is guaranteed to be at least $\geq \rho^* - \nu$. The algorithm also breaks when the solution $\mathbf{d}^{t+1}$ of the minimization problem lies at the boundary of $\mathcal{P}_N$. In this case, $\gamma_t^* = \widehat{\gamma}_t$ by an argument similar to the one used above. Thus whenever the algorithm breaks, we have $\rho^* - \nu \leq \widehat{\gamma}_t < \gamma_t^*$ and therefore TotalBoost$_\nu$ is guaranteed to output a hypothesis of margin $\gamma_t^* \geq \rho^* - \nu$.

The termination condition for LPBoost[2] follows a similar argument. Now we directly check for $\gamma_t^* \geq \widehat{\gamma}_t$. The algorithm Adaboost$_\nu^{\rho^*}$ stops as soon as its linear combination of hypotheses has margin at least $\rho^* - \nu$. Finally, Adaboost$_\nu^*$ stops as soon as the realized margin $\rho(\boldsymbol{\alpha})$ (cf. (2), using the $\boldsymbol{\alpha}$'s computed by the algorithm) is at least $\widehat{\gamma}_t \geq \rho^* - \nu$. For both of these algorithms the current distribution $\mathbf{d}^t$ lies in the interior because it is realized by finite alpha coefficients.

4. Iteration Bound

In the previous section we showed that when the algorithms break, then the output hypothesis has margin at least $\rho^* - \nu$. We now show that TotalBoost$_\nu$ must break after $T \leq \frac{2 \ln N}{\nu^2}$ iterations. In each iteration t, the algorithm updates the distribution that is "closest" to $\mathbf{d}^1$ and lies in a certain convex set and these sets get smaller as t increases. Here closeness is measured with the relative entropy which is a special Bregman divergence. This closest point is called a *projection* of $\mathbf{d}^1$ to

[2] We use a different termination condition for LPBoost than in (Bennett et al., 2000; Grove & Schuurmans, 1998).

Algorithm 2 TotalBoost$_\nu$ with accuracy param. ν

1. **Input:** $S = \langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\rangle$, desired accuracy ν
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   (a) Train classifier on $\{S, \mathbf{d}^t\}$ and obtain hypothesis $h_t : \mathbf{x} \mapsto [-1, 1]$ and let $u_n^t = y_n h_t(\mathbf{x}_n)$
   (b) Calculate the edge γ_t of h_t: $\gamma_t = \mathbf{d}^t \cdot \mathbf{u}^t$,
   (c) Set $\widehat{\gamma}_t = (\min_{q=1,\ldots,t} \gamma_q) - \nu$
   (d) Update weights:
   $$\mathbf{d}^{t+1} = \operatorname*{argmin}_{\{\mathbf{d} \in \mathcal{P}_N \,:\, \mathbf{d}\cdot\mathbf{u}^q \le \widehat{\gamma}_t, \text{ for } 1 \le q \le t\}} \Delta(\mathbf{d}, \mathbf{d}^1)$$
   (e) **If** above infeasible or $\mathbf{d}^{t+1}$ contains a zero then $T = t$ and break
4. **Output:** $f_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$, where the coefficients α_t maximize margin over hypotheses set $\{h_1, \ldots, h_T\}$.

the convex set. The proof employes the Generalized Pythagorean Theorem that holds for such projections w.r.t. any Bregman divergence.

Theorem 2 *TotalBoost$_\nu$ breaks after at most* $\lceil \frac{2 \ln N}{\nu^2} \rceil$ *iterations.*

Proof Let $\mathcal{C}_t$ denote convex set of all points $\mathbf{d} \in \mathbb{R}^N$ that satisfy $\sum_n d_n = 1$, $d_n \ge 0$ (for $1 \le n \le N$), and edge constraints $\mathbf{d} \cdot \mathbf{u}^q \le \widehat{\gamma}_t$, for $1 \le q \le t$, where $u_n^q = y_n h_q(\mathbf{x}_n)$. The distribution $\mathbf{d}^t$ at iteration $t-1$ is the projection of $\mathbf{d}^1$ onto the closed convex set $\mathcal{C}_{t-1}$. Notice that because $\widehat{\gamma}_t$ can only decrease and a new constraint is added in trial t, we have $\mathcal{C}_t \subseteq \mathcal{C}_{t-1}$. If $t \le T-1$, then our termination condition assures $\mathbf{d}^{t+1}$ exists and lies in $\mathcal{C}_{t-1}$. Furthermore, $\mathbf{d}^t$ lies in the interior of $\mathcal{P}^N$ and at trial $t-1$, there is a feasible distribution in $\mathcal{C}_{t-1} \cap \mathrm{ri}(\mathcal{P}^N)$. These two preconditions assure that the *Generalized Pythagorean Theorem* for Bregman divergences can be applied (Bregman, 1967, Lemma 1; Herbster & Warmuth, 01, Theorem 2):

$$\Delta(\mathbf{d}^{t+1}, \mathbf{d}^1) \ge \Delta(\mathbf{d}^{t+1}, \mathbf{d}^t) + \Delta(\mathbf{d}^t, \mathbf{d}^1).$$

Assume we know by induction that $\Delta(\mathbf{d}^t, \mathbf{d}^1) \ge (t-1)\frac{\nu^2}{2}$ (Base case: $t = 1$). For the inductive step we show that $\Delta(\mathbf{d}^{t+1}, \mathbf{d}^t) > \frac{\nu^2}{2}$. Observe that

$$\begin{aligned}\Delta(\mathbf{d}^{t+1}, \mathbf{d}^t) &\ge \min_{\mathbf{d} \in \mathcal{C}_t} \Delta(\mathbf{d}, \mathbf{d}^t) \\ &\ge \min_{\{\mathbf{d} \,:\, \sum_i d_n^t = 1, \mathbf{d}\cdot\mathbf{u}^t \le \gamma_t - \nu\}} \Delta(\mathbf{d}, \mathbf{d}^t), \qquad (3)\end{aligned}$$

where the second inequality follows from the fact that convex set we are minimizing over was enlarged. We form a Lagrangian for the two constraints:

$$L(\widetilde{\mathbf{d}}, \alpha, \beta) = \Delta(\widetilde{\mathbf{d}}, \mathbf{d}^t) + \alpha(\widetilde{\mathbf{d}} \cdot \mathbf{u}^t - \gamma_t + \nu) + \beta(\sum_n \widetilde{d}_n - 1).$$

By differentiating w.r.t. the primal variables $\widetilde{d}_n$ we see that the primal solution has the form:

$$d_n^* = d_n^t \exp(-\alpha u_n^t - \beta - 1).$$

Enforcing the first constraint eliminates β and the solution becomes:

$$d_n^* = \frac{d_n^t \exp(-\alpha u_n^t)}{Z(\alpha)},$$

where $Z(\alpha) = \sum_{n=1}^N d_n^t \exp(-\alpha u_n^t)$. Note that we are in iteration $t \le T-1$ and our termination condition assures all inequality constraints of $\mathcal{C}_t$ have slack and $\mathbf{d}^t \in \mathrm{ri}(\mathcal{P}^N)$. Therefore, $\mathbf{d}^* \in \mathrm{ri}(\mathcal{P}^N)$ and α must be finite. By plugging $\mathbf{d}^*$ into the Lagrangian, we get the Lagrange dual function, whose maximum over $\alpha \ge 0$ has the same value as (3) because (3) has a feasible distribution in $\mathrm{ri}(\mathcal{P}^N)$:

$$\begin{aligned}&\max_{\alpha \ge 0} \left(-(\gamma_t - \nu)\alpha - \ln Z(\alpha)\right) \\ &\quad \ge \max_{\alpha \ge 0} \left(-(\gamma_t - \nu)\alpha - \ln \widehat{Z}(\alpha)\right), \qquad (4)\end{aligned}$$

where $\widehat{Z}(\alpha) = \frac{1+\gamma_t}{2}\exp(-\alpha) + \frac{1-\gamma_t}{2}\exp(\alpha)$, is the upper bound of $Z(\alpha)$ obtained by applying the inequality $\exp(-\alpha u) \le \frac{1+u}{2}\exp(-\alpha) + \frac{1-u}{2}\exp(\alpha)$, for

Algorithm 3 TotalBoost$_\nu^{\rho^*}$ with accuracy parameter ν and maximal margin ρ^*

As TotalBoost$_\nu$ but in step 3(c) we use $\widehat{\gamma}_t = \rho^* - \nu$.

Algorithm 4 AdaBoost$_\nu^*$ with accuracy parameter ν

As TotalBoost$_\nu$ but now use minimize the divergence to the last distribution w.r.t. a single constraint:

$$\mathbf{d}^{t+1} = \operatorname*{argmin}_{\{\mathbf{d} : \mathbf{d}\cdot\mathbf{u}^t \le \widehat{\gamma}_t\}} \Delta(\mathbf{d}, \mathbf{d}^t).$$

Let α_t be the dual coefficient of the constraint on the edge of h^t used in iteration t. The algorithm breaks if the margin w.r.t. the current convex combination is at least $\widehat{\gamma}_t$.

Algorithm 5 AdaBoost$_\nu^{\rho^*}$ with accuracy parameter ν and maximal margin

As AdaBoost$_\nu^*$ but in step 3(c) we use $\widehat{\gamma}_t = \rho^* - \nu$.

$u \in [-1,1]$. By differentiating (4) we see that the last expression is minimized for $\alpha = \frac{1}{2}\ln\frac{1+\gamma_t}{1-\gamma_t} - \frac{1}{2}\ln\frac{1+\gamma_t-\nu}{1-\gamma_t+\nu}$. With this choice, (4) becomes:

$$\Delta_\pm(\gamma_t - \nu, \gamma_t) > \frac{\nu^2}{2}, \tag{5}$$

where $\Delta_\pm(a,b)$ is the binary entropy $\frac{1+a}{2}\ln\frac{1+a}{1+b} + \frac{1-a}{2}\ln\frac{1-a}{1-b}$.

The above induction shows that for $T \geq 2$

$$\Delta(\mathbf{d}^T, \mathbf{d}^1) > (T-1)\frac{\nu^2}{2} \quad \Leftrightarrow \quad T < \frac{2\ln N}{\nu^2} + 1.$$

Since T is an integer, we have $T \leq \lceil \frac{2\ln N}{\nu^2} \rceil$. ■

By scrutinizing this proof, one notices that this iteration bound also holds for the TotalBoost$^*_\nu$ algorithm because of our assumption that $\gamma_t \geq \rho^*$. The key requirement for this proof is that the closed and convex constraint sets $\mathcal{C}_t$ used for the projection at trial t must be non-increasing and contain the distributions $\mathbf{d}$ s.t. $\mathbf{d} \cdot \mathbf{u}^q \leq \rho^* - \nu$, for $1 \leq q \leq t$.

In the complete paper we prove the same iteration bound for AdaBoost$^*_\nu$, Adaboost$^{\rho^*}_\nu$, and the variants of TotalBoost where argmin$(\mathbf{d}, \mathbf{d}^1)$ is replaced by argmin$(\mathbf{d}, \mathbf{d}^t)$. The proof follows the standard approach in online learning (Kivinen & Warmuth, 1997), where the progress $\Delta(\mathbf{d}, \mathbf{d}^t) - \Delta(\mathbf{d}, \mathbf{d}^{t+1})$ is lower bounded where $\mathbf{d}$ is any comparator distribution which all constraints hold that were used up to iteration $T-1$. **Mistake: this comparator might not exist for on-line algoirthms.** Using a similar projection argument we lower bound the progress in the first $T-1$ iterations by $\frac{\nu^2}{2}$. The iteration bound is then obtained by summing the progress over the first $T-1$ iterations.

5. Experiments

In this section we illustrate the behavior of our new algorithms TotalBoost$_\nu$ & TotalBoost$^{\rho^*}_\nu$, and compare them with LPBoost and AdaBoost$^*_\nu$ on three different datasets:

- Dataset 1 is a public dataset from Telik Inc. for a drug discovery problem called COX-1: 125 binary labeled examples with a set of 3888 binary features that are complementation closed.
- Dataset 2 is an artificial dataset used in Rudin et al. (2004b) for investigating boosting algorithms that maximize the margin: 50 binary labeled examples with 100 binary features. For each original feature we added 99 similar features by inverting the feature values of 1-3 randomly chosen examples. This results in a 10,000 dimensional feature set of 100 blocks of size 100.
- Dataset 3 is a series of artificially generated datasets of 1000 examples with varying number of features but roughly constant margin. We first generated N_1 random ± 1-valued features $x_1, \ldots, x_{N_1}$ and set the label of the examples as $y = \text{sign}(x_1 + x_2 + x_3 + x_4 + x_5)$. We then duplicated each features N_2 times, perturbed the features by Gaussian noise with $\sigma = 0.1$, and clipped the feature values so that they lie in the interval [-1,1]. We considered $N_1 = 1, 10, 100$ and $N_2 = 10, 100, 1000$.

The features of our datasets represent the values of the available weak hypotheses on the examples. In each iteration of boosting, the "base learner" simply selects the feature that maximizes the edge w.r.t. the current distribution $\mathbf{d}$ on the examples. Note that our datasets and the base learner were chosen to exemplify certain properties of the algorithms and more extensive experiments are still needed.

We first discuss how the entropy minimization problems can be solved efficiently. We then compare the algorithms w.r.t. the number of iterations and the number of selected hypothesis. Finally we show how LPBoost is affected by the underlying optimizer and exhibit cases where LPBoost requires considerably more iterations than TotalBoost$_\nu$.

5.1. Solving the Entropy Problems

We use a "vanilla" *sequential quadratic programming* algorithm (Nocedal & Wright, 2000) for solving our main optimization problem:

$$\min_{\mathbf{d}\,:\,\sum_n d_n = 1,\ \mathbf{d} \geq \mathbf{0},\ \mathbf{u}^q \cdot \mathbf{d} \leq \widehat{\gamma}_t\ (1 \leq q \leq t)} \sum_{n=1}^{N} d_n \log \frac{d_n}{d_n^1}.$$

We initially set our approximate solution to $\widehat{\mathbf{d}} = \mathbf{d}^1$ and iteratively optimize $\widehat{\mathbf{d}}$. Given the current solution $\widehat{\mathbf{d}}$ satisfies the constraints $\sum_n \widehat{d}_n = 1$ and $\widehat{\mathbf{d}} \geq \mathbf{0}$, we determine an update $\boldsymbol{\delta}$ by solving the following problem:

$$\min_{\boldsymbol{\delta}} \left(\sum_{n=1}^{N} \left(1 + \log \frac{\widehat{d}_n}{d_n^1} \right) \delta_n + \frac{1}{2\widehat{d}_n} \delta_n^2 \right),$$

w.r.t. the constraints $\widehat{\mathbf{d}} + \boldsymbol{\delta} \geq \mathbf{0}$, $\sum_n \delta_n = 0$, and $\mathbf{u}^q \cdot (\widehat{\mathbf{d}} + \boldsymbol{\delta}) \leq \widehat{\gamma}_t$ (for $1 \leq q \leq t$). The estimate $\widehat{\mathbf{d}}$ is updated to $\widehat{\mathbf{d}} \leftarrow \widehat{\mathbf{d}} + \boldsymbol{\delta}$ and we repeat this process until convergence. The algorithms typically converges in very few steps.

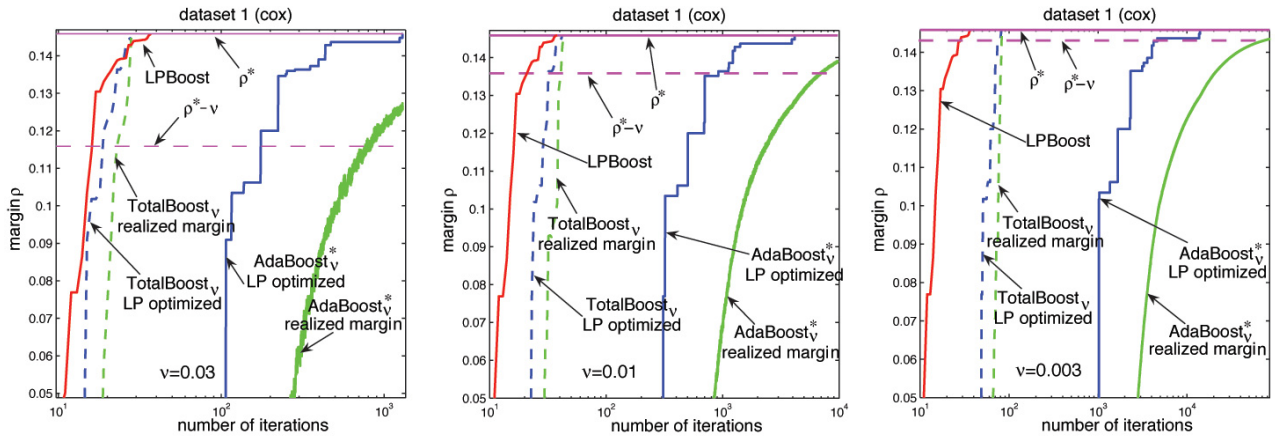

Figure 1: TotalBoost$_\nu$, LPBoost and AdaBoost$^*_\nu$ on dataset 1 for $\nu = 0.03, 0.01, 0.003$: We show the margin that is realized using the dual α coefficient of TotalBoost$_\nu$ and AdaBoost$^*_\nu$ (green) and maximum margin (blue) w.r.t. the current hypotheses set (computed with LP). Observe that AdaBoost$^*_\nu$ needs several thousands iterations and the number of iterations of TotalBoost$_\nu$ and LPBoost are comparable. The margins of TotalBoost$_\nu$ and AdaBoost$^*_\nu$ start growing slowly, in particular when ν is small. The margin of TotalBoost$^{\rho^*}_\nu$ (which knows ρ^*) increases faster than LPBoost (not shown).

Note that the above objective is the 2nd order Taylor approximation of the relative entropy $\Delta(\mathbf{d}, \mathbf{d}^1)$ at $\mathbf{d} = \widehat{\mathbf{d}}$. The resulting optimization problem is quadratic with a diagonal Hessian and can be efficiently solved by off-the-shelf optimizer packages (e.g. ILOG CPLEX).

5.2. Number of Iterations

First, we consider the number of iterations needed until each of the algorithms has achieved a margin of at least $\rho^* - \nu$. We use dataset 1 and record the margin of the convex combination of hypotheses produced by TotalBoost$_\nu$, LPBoost and AdaBoost$^*_\nu$. Additionally, we compute the maximal margin of the current hypothesis sets in each iteration. See Figure 1 for details. The default optimizer used for solving LPs and QPs is ILOG CPLEX's interior point method.

It should be noted that AdaBoost$^*_\nu$ needs considerably less computations per iteration than the totally corrective algorithms. In the case where calling the base learner is very cheap, AdaBoost$^*_\nu$ may in some unusual cases require less computation time than TotalBoost$_\nu$. However, in our experiments, the number of iterations required by AdaBoost$^*_\nu$ to achieve margin at least $\rho^* - \nu$ was $\approx 1/10$ times the theoretical upper bound $2\log(N)/\nu^2$. TotalBoost$_\nu$ typically requires much fewer iterations, even though no improved theoretical bound is known for this algorithm. In our experience, the iteration number of TotalBoost$_\nu$ depends only slightly on the precision parameter ν and when $\widehat{\gamma}_t$ is close to ρ^*, then this algorithm converges very fast to the maximum margin solution (LPBoost has a similar behavior).

While the algorithms AdaBoost$^*_\nu$ and TotalBoost$_\nu$ provably maximize the margin, they both have the problem of starting too slowly for small ν. If there is any good upper bound available for the optimal margin ρ^*, then we can initialize $\widehat{\gamma}_t$ with this upper bound and speed up the starting phase. In particular, when ρ^* is known exactly, then the algorithms AdaBoost$^{\rho^*}_\nu$ and TotalBoost$^{\rho^*}_\nu$ require drastically fewer iterations and the latter consistently beats LPBoost (not shown). In practical situations it is often easy to obtain a reasonable upper bound for ρ^*.

5.3. Number of Hypotheses

In this subsection, we compare how many hypotheses the algorithms need to achieve a large margin. Note that LPBoost and TotalBoost$_\nu$ only *select* a base hypothesis once: After the first selection, the distribution $\mathbf{d}$ is maintained such that the edge for that hypothesis is smaller than $\widehat{\gamma}_t$ and it is not selected again. AdaBoost$^*_\nu$ may select the same hypothesis many times. However, if there are several similar features (as in datasets 2 & 3), then this corrective algorithm often selects hypotheses that are similar to previously selected ones and the number of weak hypotheses used in the final convex combination is unnecessarily large. Hence, TotalBoost$_\nu$ and LPBoost seem better suited for feature selection, when small ensembles are needed.

In Figure 2 we display the margin vs. the number of *used* and *selected* hypotheses. The number of selected hypothesis for LPBoost and TotalBoost$_\nu$ is equal to the number of iterations. For these algorithms a pre-

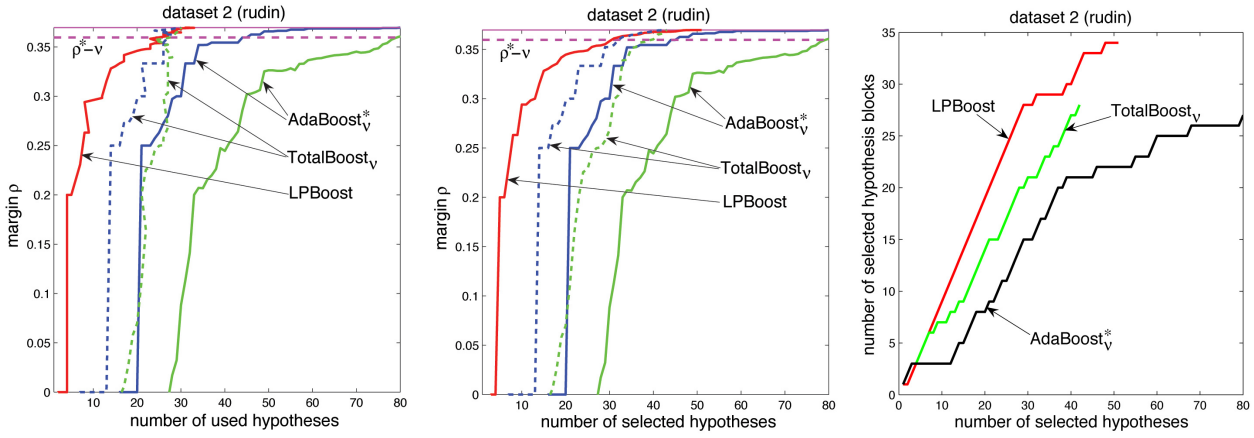

Figure 2: TotalBoost$_\nu$, LPBoost and AdaBoost$^*_\nu$ on dataset 2 for $\nu = 0.01$: [*left & middle*] We depict the realized (green) and the LP-optimized (blue) margin (as in Figure 1) versus the number of used (active) and selected (active or inactive) hypotheses in the convex combination. We observe that the totally corrective algorithms use considerable less hypotheses than the AdaBoost$^*_\nu$. If $\nu \ll 0.01$, then TotalBoost$_\nu$ is again affected by the slow start which leads to a relatively large number of selected hypotheses in the beginning. [*right*] Here we show the number of selected hypotheses versus the number of selected blocks of hypotheses. AdaBoost$^*_\nu$ often chooses additional hypotheses from a previously chosen block, while TotalBoost$_\nu$ and LPBoost typically only do this in exchange with another hypothesis from this block (not shown).

viously selected hypothesis can become inactive (corresponding $\alpha = 0$). In this case it is not counted as a used hypothesis. Note that the number of used hypotheses for LPBoost may depend on the choice of the optimizer (also see discussion below). In the case of AdaBoost$^*_\nu$, all selected hypotheses are used in the final convex combination. (See caption of Figure 2 for more details.) We can conclude that the totally corrective algorithms need considerable less hypotheses when there are many redundant hypotheses/features. LPBoost and TotalBoost$_\nu$ differ in the initial iterations (depending on ν), but produce combined hypotheses of similar size.

In Figure 3 we compare the effect of different choices of the optimizer for LPBoost. For dataset 2 there is a surprisingly large difference between interior point and simplex based methods. The reason is that the weights computed by the simplex method are often sparse and the changes in the duplicated features are sparse as well (by design). Hence, it can easily happen that the base learner is "blind" on some examples when selecting the hypotheses. Interior point methods find a solution in the interior and therefore distribute the weights among the examples. To illustrate that this is the right explanation, we modify LPBoost such that it first computes γ_t^* but then it computes the weights using the relative entropy minimization with $\widehat{\gamma}_t = \gamma_t^* + \epsilon$ (where $\epsilon = 10^{-4}$). We call this the regularized LPBoost algorithm. We observe in Figure 3 that the regularization considerably improves the speed of the simplex based solver.

5.4. Redundancy in High Dimensions

We found that LPBoost usually performs very well and is very competitive to TotalBoost$_\nu$ in terms of the number of iterations. Additionally, it only needs to solve linear and not entropy minimization problems. However, no iteration bound is known for LPBoost that is independent of the size of the hypothesis set. We performed a series of experiments with increasing dimensionality and compared LPBoost's and TotalBoost$_\nu$'s convergence speed. We found that in rather high dimensional cases, LPBoost converges quite slowly when features are redundant (see Figure 4 for an example using dataset 3). We will investigate why LPBoost converges more slowly in this example and construct more extreme datasets that show this.

6. Conclusion

We view boosting as a relative entropy projection method and obtain our iteration bounds without bounding the average training error in terms of the product of exponential potentials as is customarily done in the boosting literature (see e.g. Schapire and Singer (1999)). In the full paper we will relate our methods to the latter slightly longer proof style.

Our proof technique based on Bregman projection and the Generalized Pythagorean theorem is very versatile. The iteration bound of $O(\frac{\log n}{\nu^2})$ holds for all boosting algorithms that use constrained minimization of any Bregman divergence over a domain containing $\mathcal{P}^N$ for which $\inf_{\mathbf{d}\in\mathcal{C}_t} \widetilde{\Delta}(\mathbf{d}, \mathbf{d}^t) = \Omega(\nu^2)$ and $\widetilde{\Delta}\left(\mathbf{d}^{T-1}, (\frac{1}{n})\right) =$

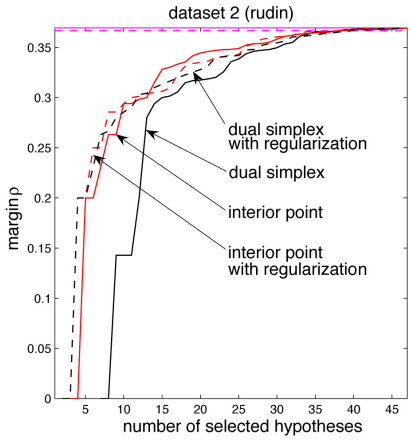

Figure 3: LPBoost with different optimizers: shown is the margin vs. the no. of selected hypotheses. Different optimizers lead to the selection of different hypotheses with varying maximum margins. Adding a regularizer (see text) significantly improves the simplex solution in some cases.

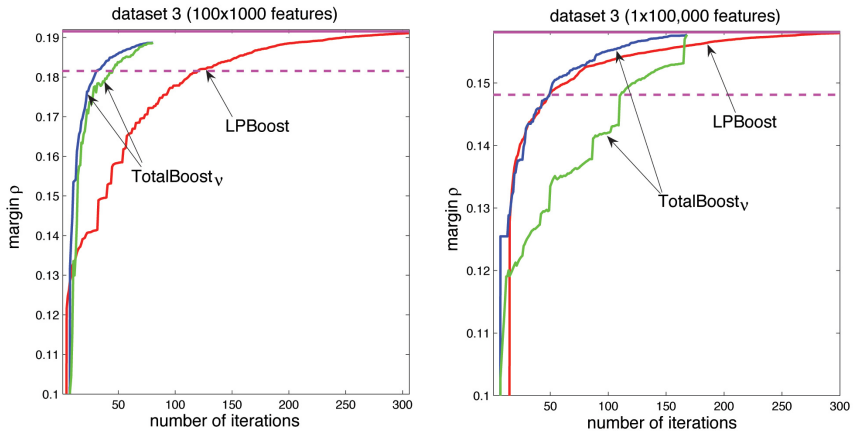

Figure 4: LPBoost vs. TotalBoost$_\nu$ on two 100,000 dimensional datasets. Shown is the margin vs. the number of iterations: [*left*] data with 100 duplicated blocks (with clipped Gaussian noise) and [*right*] data with independent features. When there are lots of duplicated features, then LPBoost stalls after an initial fast phase, while it performs well in other cases. We did not observe this behavior for TotalBoost$_\nu$ or AdaBoost$^*_\nu$ (not shown). The difference becomes larger when the block size is increased.

$O(\log n)$. For example, the sum of binary entropies Δ_2 has both these properties:

$$\inf_{\mathcal{C}_t} \overbrace{\sum_n \left(d_n \ln \frac{d_n}{d_n^t} + (1-d_n) \ln \frac{1-d_n}{1-d_n^t} \right)}^{:=\Delta_2(\mathbf{d},\mathbf{d}^t)} \geq \inf_{\mathcal{C}_t} \Delta(\mathbf{d},\mathbf{d}^t) + \underbrace{\inf_{\mathbf{d}:\sum_n d_n=1} \Delta(\mathbf{1}-\mathbf{d},\mathbf{1}-\mathbf{d}^t)}_{0} \overset{(5)}{\geq} \frac{\nu^2}{2},$$

where the first inequality follows from splitting the inf and dropping one of the constraints from the constraint set $\mathcal{C}_t$ and $\mathbf{1}$ denotes the all one vector. Furthermore, $\Delta_2\left(\mathbf{d}^{T-1}, (\frac{1}{n})\right) \leq (\ln n) + 1$ and this leads to an iteration bound of $\frac{2((\ln n)+1)}{\nu^2}$. The corrective version based on this divergence has been called *LogitBoost* (Friedman et al., 2000; Duffy & Helmbold, 2000). The above reasoning immediately provides $O(\frac{\log n}{\nu^2})$ iteration bounds for corrective and totally correction versions of LogitBoost that maximize the margin. Even though the theoretical bounds for the LogitBoost variants are essentially the same as the bounds for the standard relative entropy algorithms discussed in this paper, the LogitBoost variants are ???marginally inferior??? in practice.

Both the corrective and totally corrective algorithms for maximizing the margin start rather slowly and heuristics are needed for decreasing the edge bound $\widehat{\gamma}_t$ so that this slow start is avoided.

For practical noisy applications, boosting algorithms are needed that allow for a bias term and for soft margins. LPBoost has already been used this way in Bennett et al. (2000) but no iteration bounds are known for any version of LPBoost. We show in the full paper that our methodology still leads to iteration bounds for boosting algorithms with entropic regularization when a bias term is added. Iteration bounds for soft margin versions are left as future research.

7. Acknowledgement

The authors would thank Telik Inc. for providing the COX-1 dataset.

References

Bennett, K., Demiriz, A., & Shawe-Taylor, J. (2000). A column generation algorithm for boosting. *Proc. ICML* (pp. 65–72). Morgan Kaufmann.

Bregman, L. (1967). The relaxation method for finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Math. and Math. Physics*, *7*, 200–127.

Breiman, L. (1999). Prediction games and arcing algorithms. *Neural Computation*, *11*, 1493–1518.

Censor, Y., & Zenios, S. (1997). *Parallel optimization: Theory, algorithms and application*. Num. Math. & Sci. Comp. Oxford University Press.

Duffy, N., & Helmbold, D. (2000). Potential boosters? *NIPS* (pp. 258–264). MIT Press.

Freund, Y., & Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *J. of Comp. & Sys. Sci.*, *55*, 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, *2*, 337–374. with discussion pp.375–407, also Technical Report at Department of Statistics, Sequoia Hall, Stanford University.

Grove, A., & Schuurmans, D. (1998). Boosting in the limit: Maximizing the margin of learned ensembles. *Proc. 15th Nat. Conf. on Art. Int.*.

Herbster, M., & Warmuth, M. (01). Tracking the best linear prediction. *J. Mach. Learn. Res.*, 281–309.

Kivinen, J., & Warmuth, M. (1997). Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, *132*, 1–64.

Kivinen, J., & Warmuth, M. (1999). Boosting as entropy projection. *Proc. COLT* (pp. 134–144). ACM Press, New York, NY.

Lafferty, J. (1999). Additive models, boosting, and inference for generalized divergences. *Proc. 12th Annu. Conf. on Comput. Learning Theory* (pp. 125–133). New York, NY: ACM Press.

Nocedal, J., & Wright, S. (2000). *Numerical optimization*. Springer Series in Op. Res. Springer.

Rätsch, G., & Warmuth, M. K. (2005). Efficient margin maximizing with boosting. *J. Mach. Learn. Res.*, 2131–2152.

Rudin, C., Daubechies, I., & Schapire, R. (2004a). Dynamics of AdaBoost: Cyclic behavior and convergence of margins. *J. Mach. Learn. Res.*, 1557–1595.

Rudin, C., Schapire, R., & Daubechies, I. (2004b). Analysis of boosting algoritms using the smooth margin function: A study of three algorithms. Unpublished manuscript.

Schapire, R., Freund, Y., Bartlett, P., & Lee, W. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*, 1651–1686.

Schapire, R., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, *37*, 297–336.

von Neumann, J. (1928). Zur Theorie der Gesellschaftsspiele. *Math. Ann.*, *100*, 295–320.