Improving Algorithms for Boosting

Javed A. Aslam[*]
Department of Computer Science
Dartmouth College
6211 Sudikoff Laboratory
Hanover, NH 03755
jaa@cs.dartmouth.edu
http://www.cs.dartmouth.edu/~jaa/

Abstract

Motivated by results in information-theory, we describe a modification of the popular boosting algorithm *AdaBoost* and assess its performance both theoretically and empirically. We provide theoretical and empirical evidence that the proposed boosting scheme will have lower training and testing error than the original (non- confidence-rated) version of AdaBoost. Our modified boosting algorithm and its analysis also suggests an explanation for why boosting with confidence-rated predictions often markedly outperforms boosting without confidence-rated predictions. Finally, our motivations and analyses provide further impetus for the study of boosting in an information-theoretic, as opposed to decision-theoretic, light.

1 Introduction

Boosting is a mechanism for *training* a sequence of "weak" learners and *combining* the hypotheses generated by these weak learners so as to obtain an aggregate hypothesis which is highly accurate. One of the most popular and widely studied [11, 5, 4, 9, 10, 1, 2, 3, 7, 8] boosting algorithms is *AdaBoost*, first proposed by Freund and Schapire [5]. AdaBoost proceeds in rounds; in each round, a weak learner is *trained*, the performance of its output hypothesis is *assessed* to obtain a relative *weight* for its predictions in the output combination, and the distribution is *modified* for the next boosting round according to this weight. AdaBoost essentially assesses the performance of weak hypotheses by measuring the overall *error* of the hypothesis.

In this paper, we argue that while predictive error is most often used to assess the overall performance of learning algorithms, it is not necessarily the best measure of performance for weak hypotheses in a boosting scenario. We argue that the true performance of a weak hypothesis is *captured* by the joint distribution of its predictions with the correct labels, and that this performance is best *measured* for the purposes of boosting by the *odds* of making a correct prediction *given* a particular output label or by such information-theoretic measures as mutual information or conditional entropy.

Based on these ideas, we propose a modification of the AdaBoost algorithm and analyze its performance both theoretically and empirically. We derive bounds on the training error of the proposed boosting algorithm and give evidence that its training error is lower than that of AdaBoost. Our analysis yields a criterion for determining the best weak hypothesis to use in any round of boosting in order to minimize training error.

For the case of non- confidence-rated hypotheses, we demonstrate that AdaBoost does not generally make good use of the best weak hypotheses as dictated by information-theory, while our proposed modified boosting algorithm does. For the case of confidence-rated hypotheses, there is a strong connection between our proposed modified boosting algorithm and techniques suggested by Schapire and Singer [11]. Our analysis suggests an information-theoretic explanation for why boosting with confidence-rated predictions markedly outperforms boosting without confidence-rated predictions in many cases. Together with other recent analyses (for example, Kivinen and Warmuth [6]), this provides further impetus for the study of boosting in an information-theoretic, as opposed to decision-theoretic, light.

Finally, we describe a number of experiments using UCI and TREC data sets.

2 Motivation

Consider the version of AdaBoost proposed by Schapire and Singer [11] as shown in Figure 1. Here, the sequence (x_1, y_1), (x_2, y_2), ... are *training examples* where each x_i is an element of a known *instance space* $\mathcal{X}$ (*e.g.*, $\{0,1\}^n$, $\mathbb{R}^n$, etc.) and each y_i is an element of a known *label space* $\mathcal{Y}$ (*e.g.*, $\{0,1\}$, $\{1,\ldots,k\}$, etc.). The version of AdaBoost given in Figure 1 is concerned with the *binary* classification problem and for mathematical convenience uses a $\{-1,+1\}$ label space.

Like most boosting algorithms, AdaBoost proceeds in rounds. In each round t, a weak learner is trained using a local distribution D_t (initially uniform) over the training examples, and a weak hypothesis h_t is returned. Note that h_t is assumed to have range $\mathbb{R}$; the *sign* of h_t is interpreted as the desired *label*, and the *magnitude* of h_t is interpreted as a *confidence* in this label assessment. Such hypotheses are referred to as having *confidence-rated* predictions. When hypotheses are restricted to have range $\{-1,+1\}$, then this version of AdaBoost is identical to Freund and Schapire's

[*]This work partially supported by NSF Grant EIA-98-02068.

Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

1. Train weak learner using distribution D_t.
2. Get weak hypothesis $h_t : \mathcal{X} \to \mathbb{R}$.
3. Choose $\alpha_t \in \mathbb{R}$.
4. Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Figure 1: Schapire and Singer's AdaBoost [11]

original version [5].

Next, the performance of h_t with respect to the training distribution is assessed, and this performance is used in determining the α_t parameter used in the distribution update rule; the "better" the performance of h_t, the "larger" a value of α_t will be chosen. The update rule then modifies the distribution for training in the next round: correctly classified examples are decreased in weight and incorrectly classified examples are increased in weight. The larger α_t, and hence the better the performance of h_t, the more the distribution is "skewed." Finally, the output hypothesis is a linear combination of the outputs of the respective weak hypotheses where the weight associated with a weak hypothesis is its performance assessment α_t.

For the purposes of motivating our proposed algorithm for boosting, let us restrict ourselves for the moment to hypotheses with $\{-1, +1\}$ range. Consider drawing examples (x_i, y_i) at random from the training set according to the distribution D, and let Y and H be random variables corresponding to the labels, y_i, and predictions, $h(x_i)$, respectively. We may then consider the *joint distribution* $p_D(Y, H)$ where $p_D(Y = y, H = h)$ is the probability of drawing an example (x_i, y_i) according to D where $y_i = y$ and $h(x_i) = h$; we denote this as follows

$$p_D(Y = y, H = h) = \Pr_{i \sim D_t}[y_i = y, h(x_i) = h].$$

Note that the four probabilities

$$\begin{array}{l} p_D(Y = -1, H = -1), \\ p_D(Y = -1, H = +1), \\ p_D(Y = +1, H = -1), \text{ and} \\ p_D(Y = +1, H = +1) \end{array}$$

are simply the weight of true negative, false positive, false negative and true positive predictions. Finally, note that the

h_1	-1	$+1$
-1	$5/16$	$3/16$
$+1$	$1/16$	$7/16$

h_2	-1	$+1$
-1	$6/16$	$2/16$
$+1$	$2/16$	$6/16$

Figure 2: Joint distributions $p_D(Y, H_1)$ and $p_D(Y, H_2)$. Rows correspond to correct labels and columns correspond to predictions.

error of a hypothesis is simply the sum of the false positive and false negative weights; *i.e.*,

$$\begin{array}{l} \text{error}_D(h) = \\ \qquad p_D(Y = -1, H = +1) + p_D(Y = +1, H = -1). \end{array}$$

Schapire and Singer [11] have shown that the performance of AdaBoost is optimized when α_t is set as follows. Let $\epsilon_t = \text{error}_{D_t}(h_t)$, then $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$. Thus, the "quality" of the predictor h_t is (one-half) the log-odds of h_t making a correct prediction. For this setting of α_t, the distribution update rule has the effect of uniformly scaling correctly classified examples by a $\sqrt{\epsilon_t/(1-\epsilon_t)}$ factor; similarly, incorrectly classified examples are uniformly scaled by a $\sqrt{(1-\epsilon_t)/\epsilon_t}$ factor. The net effect of this scaling ensures that $\text{error}_{D_{t+1}}(h_t) = 1/2$; in other words, Y and H_t are "uncorrelated" with respect to error for the distribution D_{t+1}.

The motivation for our proposed boosting algorithm is the fact that the qualitative and quantitative performance of a predictor h is not entirely captured by error. The performance *is* captured by the joint distribution $p_D(Y, H)$, and various measures of performance can be calculated from this joint distribution, where error is but one such measure. Consider, for example, two predictors h_1 and h_2 and their respective joint distributions, $p_D(Y, H_1)$ and $p_D(Y, H_2)$, as shown in Figure 2. While h_1 and h_2 have the same error, 25%, they are quite different qualitatively. The hypothesis h_1 has a lower false negative rate at the expense of a higher false positive rate. From an information-theoretic perspective, more information about the correct label is gained when h_1 predicts -1 than when h_1 predicts $+1$. In fact, this information gain can be *quantified*; in bits, it is the entropy of Y minus the conditional entropy of Y given h_1, $H_2(Y) - H_2(Y|H_1 = h_1)$. In fact, the expected information gained about the correct label given a prediction is precisely the mutual information of Y and H_1, $I(Y, H_1)$; we strongly believe that weak hypothesis should be optimized with respect to this measure, and experiments with our proposed boosting algorithm bear out the belief.

Now consider the *odds* of a correct prediction *given the actual prediction made*. When h_1 predicts -1, its odds of being correct are 5:1; when h_1 predicts $+1$, its odds of being correct are 7:3. For h_2, both odds are 3:1, and this corresponds to the overall odds of being correct for both hypotheses as well. Thus, negative predictions by h_1 are more trustworthy than positive predictions, and we can *assess* this relative trustworthiness using the joint distribution. We propose that such knowledge should be used by the boosting algorithm itself: whereas AdaBoost assigns an *overall* weight α_t

Given: $(x_1, y_1), \ldots, (x_m, y_m); x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

1. Train weak learner using distribution D_t.
2. Get weak hypothesis $h_t : \mathcal{X} \to \mathbb{R}$.
3. Choose $\alpha_t[-1] \in \mathbb{R}$ and $\alpha_t[+1] \in \mathbb{R}$; let $\alpha_t(z) = \begin{cases} \alpha_t[-1] & \text{if } z < 0, \\ \alpha_t[+1] & \text{if } z \geq 0. \end{cases}$
4. Update:
$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t(h_t(x_i)) y_i h_t(x_i))}{Z_t}$$
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:
$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t(h_t(x)) h_t(x)\right).$$

Figure 3: InfoBoost

to the predictions of any given weak hypothesis h_t, our proposed boosting algorithm assigns *individual* weights to each of the possible labels predicted by h_t.

When hypothesis predictions are restricted to the range $\{-1, +1\}$, the weight assigned to h_t by AdaBoost is proportional to the log-odds that h_t will make a correct prediction. In the proposed modification to AdaBoost, we derive the optimal weights that should be assigned individually to positive and negative predictions, and we provide evidence, both theoretical and empirical, that the proposed algorithm will have lower training and testing error.

When hypothesis predictions are allowed to be confidence-rated, our techniques provide an alternate derivation of the weights Schapire and Singer [11] assign to 2-block domain-partitioning hypotheses. Our analysis further suggests an *information-theoretic explanation* for why these weights are good and hence why the use of confidence-rated hypotheses may out-perform the use of non- confidence-rated hypotheses.

3 InfoBoost

In this section, we describe and analyze our proposed modification of AdaBoost, as motivated in the previous section. Our descriptions and analyses are modeled closely on those used by Schapire and Singer [11]. Consider the proposed boosting algorithm shown in Figure 3; for purposes of distinction, we refer to this boosting algorithm as *InfoBoost*. InfoBoost is mechanically quite similar to AdaBoost except in three important regards. First, in each round t of boosting, *two* performance parameters are calculated or chosen: $\alpha_t[-1]$ and $\alpha_t[+1]$. As motivated in the previous section, $\alpha_t[-1]$ is a function of the quality of a negative prediction, and $\alpha_t[+1]$ is a function of the quality of a positive prediction. Second, the distribution update rule effectively recognizes *four* classes of examples (false positive, false negative, true positive, and true negative), and the update rule scales the weight of any example according to its class. Third, the final hypothesis weights the outputs of the individual weak hypotheses differently, depending on their actual prediction values.

3.1 Analysis

We now analyze the performance of InfoBoost and derive optimal settings for the $\alpha_t[\cdot]$ parameters; again, our analysis is closely modeled on that of Schapire and Singer [11]. We first calculate the final distribution over the examples by unfolding the distribution update recurrence

$$\begin{aligned} D_{T+1}(i) &= \frac{D_T(i) \exp(-\alpha_t(h_t(x_i)) y_i h_t(x_i))}{Z_t} \\ &= \frac{\exp(-\sum_t \alpha_t(h_t(x_i)) y_i h_t(x_i))}{m \prod_t Z_t}. \end{aligned}$$

Note that the exponent in the numerator of this quantity is quite similar to the final hypothesis itself. Letting $f(x) = \sum_t \alpha_t(h_t(x)) h_t(x)$, we have $H(x) = \text{sign}(f(x))$ and

$$D_{T+1}(i) = \frac{\exp(-y_i f(x_i))}{m \prod_t Z_t}. \quad (1)$$

Let $[\pi]$ be an indicator variable which is 1 if the predicate π is true and 0 otherwise. We then have the following sequence of implications.

$$\begin{aligned} H(x_i) \neq y_i \quad &\Rightarrow \quad y_i f(x_i) \leq 0 \\ &\Rightarrow \quad \exp(-y_i f(x_i)) \geq 1 \\ &\Rightarrow \quad [H(x_i) \neq y_i] \leq \exp(-y_i f(x_i)), \text{ and} \\ H(x_i) = y_i \quad &\Rightarrow \quad y_i f(x_i) \geq 0 \\ &\Rightarrow \quad \exp(-y_i f(x_i)) \geq 0 \\ &\Rightarrow \quad [H(x_i) \neq y_i] \leq \exp(-y_i f(x_i)). \end{aligned}$$

Thus, in all cases we have $[H(x_i) \neq y_i] \leq \exp(-y_i f(x_i))$. To bound the weight of misclassified examples, we need only sum our indicator variable over all examples and divide by m, as the initial distribution was uniform. Applying the above fact and Equation 1, we have the following.

$$\begin{aligned} \frac{1}{m} \sum_i [H(x_i) \neq y_i] &\leq \frac{1}{m} \sum_i \exp(-y_i f(x_i)) \\ &= \frac{1}{m} \sum_i \left(m\, D_{T+1}(i) \prod_t Z_t \right) \\ &= \prod_t Z_t \end{aligned}$$

We thus have the following result analagous to AdaBoost.

Theorem 1 *The training error of InfoBoost is at most*

$$\text{error}_D(H(x)) \leq \prod_{t=1}^{T} Z_t.$$

3.2 Optimizing $\alpha[-1]$ and $\alpha[+1]$

Now, in order to optimize the performance of InfoBoost, we must ensure that Z_t is as small as possible in each round of boosting. In round t, we have

$$Z_t = \sum_{i=1}^{m} D_t(i)\exp(-\alpha_t(h_t(x_i))y_i h_t(x_i)).$$

Following Schapire and Singer [11], let us fix t and set $u_i = y_i h_t(x_i)$. We will allow confidence-rated predictions, though restrict their range to $[-1,+1]$. We then have

$$\begin{aligned} Z &= \sum_{i:h(x_i)<0} D(i)e^{-\alpha[-1]\cdot u_i} + \sum_{i:h(x_i)\geq 0} D(i)e^{-\alpha[+1]\cdot u_i} \\ &\leq \sum_{i:h(x_i)<0}\left(\frac{1+u_i}{2}e^{-\alpha[-1]} + \frac{1-u_i}{2}e^{\alpha[-1]}\right) + \\ &\quad \sum_{i:h(x_i)\geq 0}\left(\frac{1+u_i}{2}e^{-\alpha[+1]} + \frac{1-u_i}{2}e^{\alpha[+1]}\right). \end{aligned} \quad (2)$$

Solving the equations $\partial Z/\partial\alpha[-1] = 0$ and $\partial Z/\partial\alpha[+1] = 0$, we find that Z is minimized when

$$\begin{aligned} \alpha[-1] &= \frac{1}{2}\ln\left(\frac{1+r[-1]}{1-r[-1]}\right) \\ \alpha[+1] &= \frac{1}{2}\ln\left(\frac{1+r[+1]}{1-r[+1]}\right) \end{aligned}$$

and where

$$\begin{aligned} r[-1] &= \frac{\sum_{i:h(x_i)<0} D(i)u_i}{\sum_{i:h(x_i)<0} D(i)} \\ r[+1] &= \frac{\sum_{i:h(x_i)\geq 0} D(i)u_i}{\sum_{i:h(x_i)\geq 0} D(i)}. \end{aligned}$$

Plugging these optimal values of $\alpha[\cdot]$ into Equation 2, we have

$$\begin{aligned} Z &\leq \Pr_D[h<0]\cdot\sqrt{1-r[-1]^2} + \\ &\quad \Pr_D[h\geq 0]\cdot\sqrt{1-r[+1]^2} \end{aligned} \quad (3)$$

where

$$\begin{aligned} \Pr_D[h<0] &= \sum_{i:h(x_i)<0} D(i), \text{ and} \\ \Pr_D[h\geq 0] &= \sum_{i:h(x_i)\geq 0} D(i). \end{aligned}$$

Theorem 2 *For weak hypotheses whose range is $[-1,+1]$, the training error of InfoBoost is at most*

$$\begin{aligned} \mathrm{error}_D(H(x)) \;\leq\; \prod_{t=1}^{T}\Big(&\Pr_{D_t}[h_t<0]\cdot\sqrt{1-r_t[-1]^2} + \\ &\Pr_{D_t}[h_t\geq 0]\cdot\sqrt{1-r_t[+1]^2}\Big). \end{aligned}$$

Finally, for confidence-rated hypotheses whose range is outside $[-1,+1]$, we note that the optimal settings for $\alpha_t[\cdot]$ can be determined numerically, in a manner similar to that suggested by Schapire and Singer [11].

3.3 Comparison to AdaBoost

These results compare favorably to results obtained for AdaBoost. Schapire and Singer [11] show that for AdaBoost,

- $\mathrm{error}_D(H'(x)) \leq \prod_{t=1}^{T} Z'_t$ and
- $Z' \leq \sqrt{1-r^2}$ where $r = \sum_i D'(i)u_i$.

Now consider fixing a local distribution D and running one boosting round of both AdaBoost and InfoBoost with the same weak learner; let h be the hypothesis returned by the weak learner. We note that $r = \Pr_D[h<0]\cdot r[-1] + \Pr_D[h\geq 0]\cdot r[+1]$. Furthermore, since $\sqrt{1-x^2}$ is a concave down function, by a simple convexity argument we have

$$\begin{aligned} \sqrt{1-r^2} &\geq \Pr_D[h<0]\cdot\sqrt{1-r[-1]^2} + \\ &\quad \Pr_D[h\geq 0]\cdot\sqrt{1-r[+1]^2}. \end{aligned} \quad (4)$$

The inequality above is strict, unless $r[-1] = r[+1]$ or either $\Pr_D[h<0]$ or $\Pr_D[h\geq 0]$ is zero. Thus, we may conclude that the upper bound on Z for InfoBoost is at most the upper bound on Z for AdaBoost for *any* weak hypothesis, even those optimized for AdaBoost. Of course, users of AdaBoost would reasonably seek weak learners that optimize $\sqrt{1-r^2}$, while users of InfoBoost would reasonably seek weak learners that optimize the expression on the right-hand side of Equation 3. In this case, the difference in these bounds would likely be even greater. We note, however, that it is difficult to fairly compare the performance of AdaBoost and InfoBoost since, in the above discussion, we are comparing *upper bounds*, and in general, the distributions used by these boosters in any round t will in all likelihood be different.

Finally, the effect seen in Equation 4 is particularly easy to interpret when using hypotheses restricted to the range $\{-1,+1\}$ (*i.e.*, hypotheses without confidence ratings). In this case, we have

$$\begin{aligned} \sqrt{1-r^2} &= 2\sqrt{\epsilon(1-\epsilon)} \\ \sqrt{1-r[-1]^2} &= 2\sqrt{\epsilon[-1](1-\epsilon[-1])} \\ \sqrt{1-r[+1]^2} &= 2\sqrt{\epsilon[+1](1-\epsilon[+1])} \end{aligned}$$

where ϵ is the *error* of the weak hypothesis, $\epsilon[-1]$ is the *conditional error* of the hypothesis given the prediction -1, and $\epsilon[+1]$ is the *conditional error* of the hypothesis given the prediction $+1$. Thus, the bound on Z for AdaBoost is $2\sqrt{\epsilon(1-\epsilon)}$, and the bound on Z for InfoBoost is

$$\begin{aligned} &\Pr_D[h<0]\cdot 2\sqrt{\epsilon[-1](1-\epsilon[-1])} + \\ &\quad \Pr_D[h\geq 0]\cdot 2\sqrt{\epsilon[+1](1-\epsilon[+1])}. \end{aligned}$$

The plot in Figure 4 illustrates the difference in these bounds.

3.4 An Information-Theoretic Perspective

As motivated earlier, we believe that weak learners which *maximize* mutual information (equivalently, *minimize* conditional entropy) should be used in learning and boosting — by maximizing mutual information, the uncertainty (entropy) of the correct label is reduced as much as possible, in

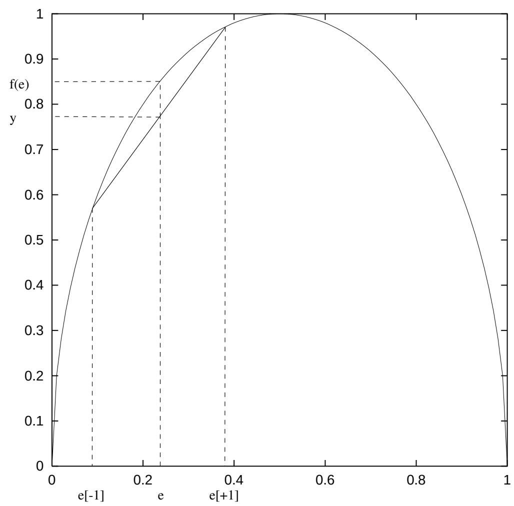

Figure 4: This is a plot of the function $2\sqrt{x(1-x)}$. On the x-axis, e is the error of a hypothesis, while $e[-1]$ and $e[+1]$ are the respective conditional errors given the predictions -1 and $+1$, respectively. On the y-axis, $f(e) = 2\sqrt{e(1-e)}$ is the bound on Z for AdaBoost, and y is the bound on Z for InfoBoost.

h_t	-1	$+1$
-1	$5/16$	$3/16$
$+1$	$1/16$	$7/16$

h_t	-1	$+1$
-1	$5/24$	$3/8$
$+1$	$1/8$	$7/24$

h_t	-1	$+1$
-1	$\frac{\sqrt{5}/2}{\sqrt{5}+\sqrt{21}}$	$\frac{\sqrt{21}/2}{\sqrt{5}+\sqrt{21}}$
$+1$	$\frac{\sqrt{5}/2}{\sqrt{5}+\sqrt{21}}$	$\frac{\sqrt{21}/2}{\sqrt{5}+\sqrt{21}}$

Figure 5: From top to bottom, joint distributions $p_{D_t}(Y, H_t)$, $p_{D_{t+1}}(Y, H_t)$ for the AdaBoost update rule, and $p_{D_{t+1}}(Y, H_t)$ for the InfoBoost update rule. Rows correspond to correct labels and columns correspond to predictions. Note that the joint distribution produced by the AdaBoost update rule is not a product distribution; thus, H_t and Y are not independent with respect to D_{t+1}. The joint distribution produced by the InfoBoost update rule is a product distribution, however.

an information-theoretic sense. In this section, we argue that InfoBoost can make good use of such weak learners, while the non- confidence-rated version of AdaBoost, in general, can not. Our experimental results given in the next section bear this out as well.

Consider hypotheses which are not confidence-rated; *i.e.*, $h : \mathcal{X} \Rightarrow \{-1, +1\}$. As discussed in Section 2, the effect of AdaBoost's update rule is to create a *new* distribution D_{t+1} such that the predictions of the *previous* hypothesis are uncorrelated with the correct labels using the error measure; *i.e.*, $\mathrm{error}_{D_{t+1}}(h_t) = 1/2$. However, we believe that there are much better measures of correlation than error. Using the notation described in Section 2, consider the joint distribution $p_{D_{t+1}}(Y, H_t)$. We propose using either probabilistic or information-theoretic measures of correlation. In fact, when considering complete independence, these measures are equivalent: two random variables are probabilistically independent, and their mutual information is zero, precisely when their joint distribution is a product of marginal distributions. The AdaBoost update rule does *not* guarantee independence, while the InfoBoost update rule *does* guarantee the probabilistic and information-theoretic independence of Y and H_t with respect to D_{t+1}. We argue the former by example in Figure 5; we argue the latter as follows. For hypotheses which are not confidence-rated, the optimal settings for $\alpha[-1]$ and $\alpha[+1]$ are as follows

$$\alpha[-1] = \frac{1}{2}\ln\left(\frac{1-\epsilon[-1]}{\epsilon[-1]}\right) \quad (5)$$

$$\alpha[+1] = \frac{1}{2}\ln\left(\frac{1-\epsilon[+1]}{\epsilon[+1]}\right). \quad (6)$$

Thus, $\alpha[-1]$ is one-half the log odds of being correct when predicting -1, and $\alpha[+1]$ is one-half the log odds of being correct when predicting $+1$. The effect of the InfoBoost update rule is to ensure that these odds are 1:1; *i.e.*, the InfoBoost update rule ensures that the weights of true negative and false negative examples are identical and that the weights of true positive and false positive examples are identical. One can easily show that any joint distribution satisfying these requirements must be a product of its marginals and that the error associated with such a joint distribution is 1/2. Thus, the InfoBoost update rule guarantees that with respect to D_{t+1}, the predictions of h_t are uncorrelated with the correct labels in a probabilistic, information-theoretic and error-based sense.

The effect of these facts in practice is as follows. It is possible that the weak hypothesis which maximizes mutual information has an error rate of 1/2. In this situation, AdaBoost will cease to improve: the AdaBoost update rule will not modify the distribution over examples, and the same weak hypothesis will be chosen in subsequent rounds. In our experiments detailed in the following section, we see this effect with AdaBoost regularly and quite early, typically by the 50th boosting round in our experiments. However, this cannot happen when using InfoBoost since the previous hypothesis always has a mutual information of zero with respect to the current distribution. In fact, InfoBoost most often performs best in practice with a weak learner which maximizes mutual information; these results are given in the next section.

Finally, we note that when confidence-rated hypotheses are allowed, the $\alpha[\cdot]$ *weights* derived in Equations 5 and 6 are identical to the *confidences* assigned by Schapire and

Singer [11] to an analogous 2-block domain-partitioning hypothesis. Thus, one probable explanation for why such confidence-rated predictions can markedly outperform non- confidence-rated predictions is that these confidence ratings, together with their use in the distribution update rule, guarantee that, with respect to the *subsequent* distribution, the *current* (weak) hypothesis will be uncorrelated with the correct labels in the strongest possible (information-theoretic) sense. Note that simply being uncorrelated in an error-based sense is a much weaker condition.

4 Experiments

As we were initially motivated in this work by the TREC (Text Retrieval Conference) batch filtering task, we conducted a number of experiments using TREC data, and they are presented in the next section. Experiments comparing AdaBoost and InfoBoost using the UCI machine learning datasets are currently being conducted; preliminary results for the Monk data sets are presented in a subsequent section.

4.1 TREC Experiments

In the TREC filtering task, a large collection of *text documents* are provided, together with a collection of *topics* and *relevance judgments*. Each topic is a specification of interest (*e.g.*, "The downfall of communism in Eastern Europe"), and for each topic, relevance assessments for a subset of the given documents are provided.

We ran our experiments using data from TREC7. In this competition, the document collection was the entire set of AP newswire articles from 1988, 1989 and 1990; relevance judgments were provided for 50 topics. Many of the topics in this competition were anomalous; for example, a number of topics had only a handful, or even just one, judged relevant document. We eliminated those topics which had fewer than 150 judged relevant documents and conducted our experiments for each of the remaining 18 topics using only those documents for which relevance assessments were provided. As the learning curves were similar across topics, we report average prediction errors over all topics.

We created weak hypotheses from words as follows. For any word in the collection, a weak hypothesis corresponding to this word predicts $+1$ if the word appears in a given document and -1 otherwise. We also considered anti-word predictors which are identical to their corresponding word predictors except in the signs of their predictions.[1] We then devised three weak learners:

1. *Error:* Returns the weak hypothesis which minimizes the usual prediction error with respect to the given distribution. This weak learner is optimized for AdaBoost in the sense described in Section 3.3.

2. *Opt:* Returns the weak hypothesis which minimizes, with respect to the given distribution, the expression given in the right-hand side of Equation 3. This weak learner is optimized for InfoBoost in the sense described in Section 3.3.

3. *Mutual Information:* Returns the weak hypothesis which maximizes mutual information with respect to the given distribution.

In the TREC7 competition, training occurred over the 1989 AP collection, and testing occurred over the 1988 and 1990 collections. We conducted five boosting experiments over each of the 18 topics:

1. AdaBoost with the Error weak learner.

2. AdaBoost with the Mutual Information weak learner.

3. InfoBoost with the Error weak learner.

4. InfoBoost with the Opt weak learner.

5. InfoBoost with the Mutual Information weak learner.

Our results are given in Figure 6. With respect to *training error*, we note that after about 50 rounds of boosting, AdaBoost with the Mutual Information weak learner ceases to make any progress. This in all likelihood is due to the reasons outlined in Section 3.3. Of the remaining four combinations, AdaBoost with the Error weak learner performed *worst*, followed by InfoBoost with the Error weak learner, InfoBoost with the Mutual Information weak learner, and InfoBoost with the Opt weak learner which performed *best*.

With respect to *testing error*, AdaBoost with the Mutual Information weak learner again ceased to improve after roughly 50 rounds of boosting; its performance was much worse than the others, leveling off at about 28% error.[2] Of the remaining four combinations, AdaBoost with the Error weak learner performed *worst*, followed by InfoBoost with the Error weak learner. InfoBoost with the Opt weak learner and InfoBoost with the Mutual Information weak learner performed *best*; their performance was virtually indistinguishable after 300 rounds of boosting.

4.2 UCI Experiments

The Monk data sets, available at the University of California at Irvine's machine learning repository,[3] are particularly simple with which to experiment (test sets are provided, all attributes are discrete, and all labels are binary). We present preliminary results for these data sets in Figures 7, 8 and 9 below. (Note that the performance of InfoBoost with the Opt weak learner and InfoBoost with the Mutual Information weak learner were essentially identical; their learning curves are quite often superimposed.) Particularly interesting are the results from the Monk 3 data set, wherein 5% classification noise was deliberately added to the training set. On this noisy data set, InfoBoost's test error was substantially lower than AdaBoost's.

[1] Strictly speaking, anti-words are unnecessary, since both AdaBoost and InfoBoost can make use of predictors which are anti-correlated.

[2] We note that the trivial hypothesis which always predicts -1 had an average training and testing error of 29%; thus, the combination of AdaBoost with the Mutual Information weak learner had negligible performance.

[3] http://www.ics.uci.edu/~mlearn/MLRepository.html

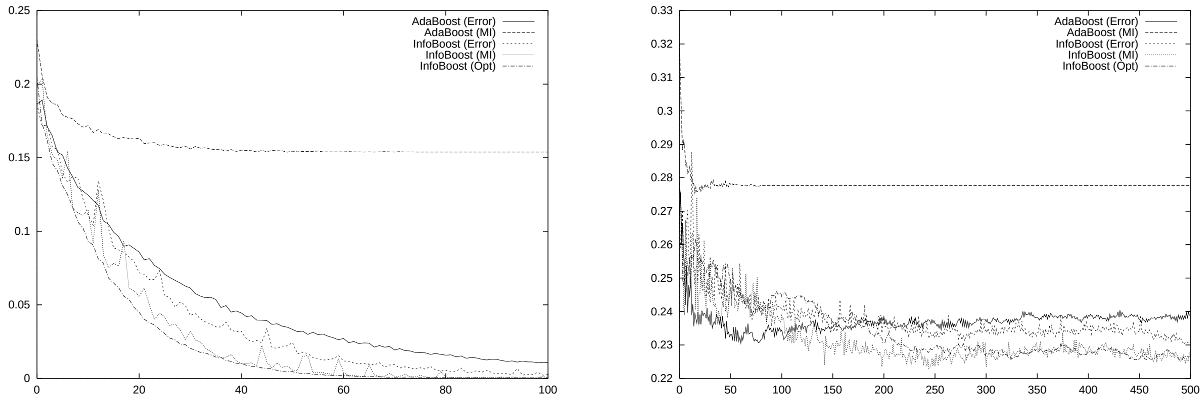

Figure 6: Average training and testing errors of five combination systems on TREC filtering data. The left plot is training error, and the right plot is testing error. The x-axis corresponds to boosting round, and the y-axis corresponds to average error over 18 topics.

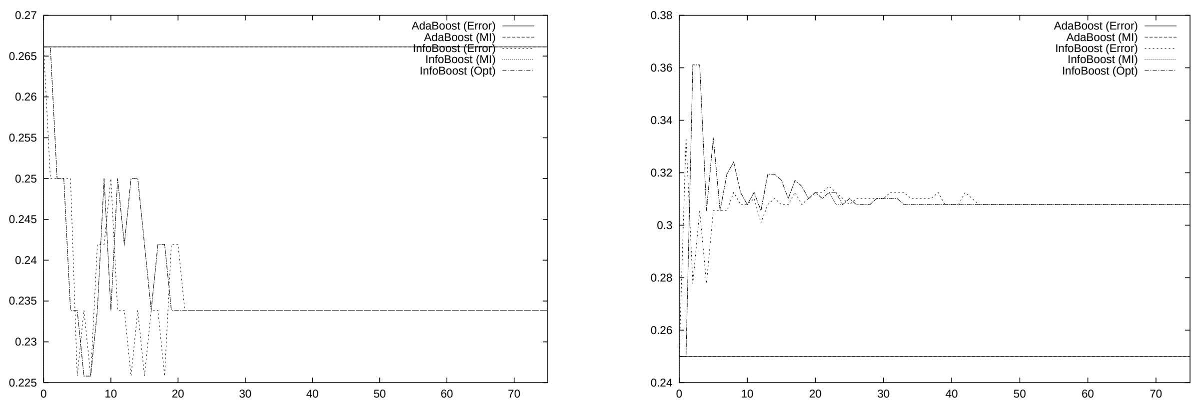

Figure 7: Average training and testing errors of five combination systems on the Monk 1 data set. The left plot is training error, and the right plot is testing error. The x-axis corresponds to boosting round, and the y-axis corresponds to prediction error.

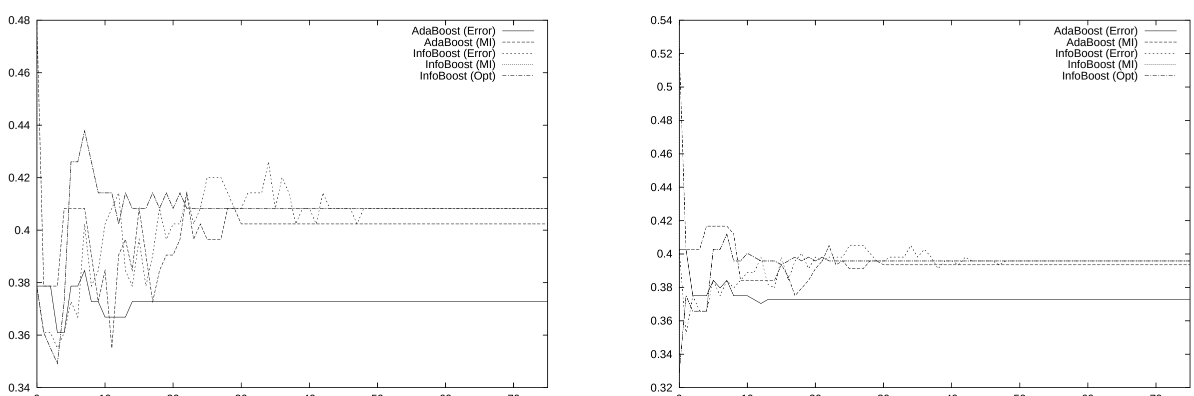

Figure 8: Average training and testing errors of five combination systems on the Monk 2 data set. The left plot is training error, and the right plot is testing error. The x-axis corresponds to boosting round, and the y-axis corresponds to prediction error.

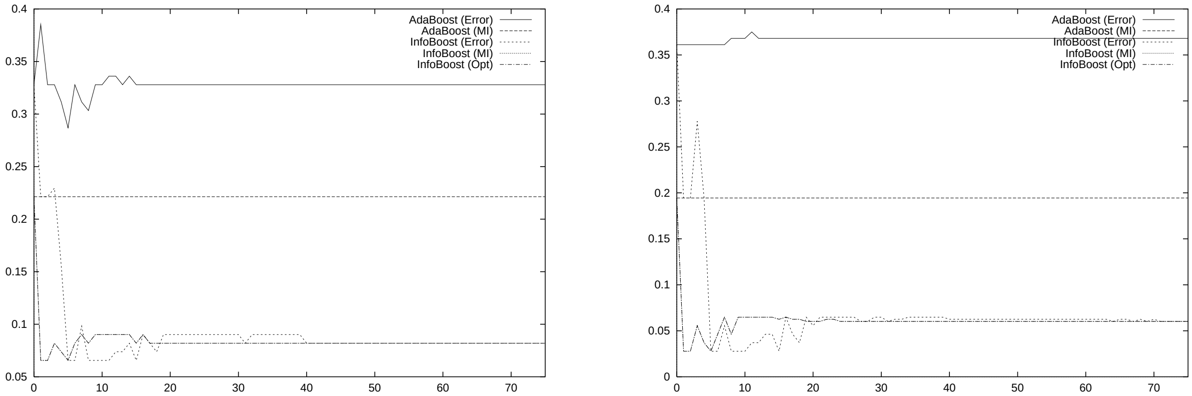

Figure 9: Average training and testing errors of five combination systems on the Monk 3 data set. The left plot is training error, and the right plot is testing error. The x-axis corresponds to boosting round, and the y-axis corresponds to prediction error.

5 Conclusions

We have described a modification of AdaBoost and assessed its performance, both theoretically and empirically. We have provided theoretical and empirical evidence that the proposed boosting scheme will have lower training and testing error than the original (non- confidence-rated) version of AdaBoost. Our modified boosting algorithm and its analysis also suggests an *information-theoretic explanation* for why boosting with confidence-rated predictions often markedly outperforms boosting without confidence-rated predictions. We strongly feel that weak hypotheses should be chosen based on information-theoretic criteria and that boosting algorithms themselves should be designed around information-theoretic principles. Together with other recent results suggesting that AdaBoost can be analyzed information-theoretically [6], we hope to provide further impetus for the study of boosting in an information-theoretic, as opposed to decision-theoretic, light.

Acknowledgements

I would like to thank Mark Montague and Dan Scholnick for conducting the experiments given in Section 4 and David Latham for implementing much of our system for experiments.

References

[1] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Unpublished manuscript, 1997.

[2] Leo Breiman. Arching classifiers. *Annals of Statistics*, to appear.

[3] Harris Drucker and Corinna Cortes. Boosting decision trees. *Advances in Neural Information Processing Systems*, (8):479–485, 1996.

[4] Yoav Freund and Robert Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996.

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

[6] Jyrki Kivinen and Manfred K. Warmuth. Boosting as entropy projection. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, 1999.

[7] Richard Maclin and David Opitz. An empirical evaluation of bagging and boosting. In *Prodeedings of the Fourteenth National Conference on Artificial Intelligence*, 1997.

[8] J. R. Quinlan. Bagging, boosting, and c4.5. In *Prodeedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.

[9] Robert Schapire. Using output codes to boost multiclass machine learning problems. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.

[10] Robert Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: a new explanation for the effectiveness of the voting method. *Annals of Statistics*, to appear.

[11] Robert Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, 1998.