AR-Boost: Reducing Overfitting by a Robust Data-Driven Regularization Strategy

Baidya Nath Saha[1], Gautam Kunapuli[2], Nilanjan Ray[3], Joseph A. Maldjian[1], and Sriraam Natarajan[4]

[1] Wake Forest School of Medicine, USA
{bsaha,maldjian}@wakehealth.edu
[2] University of Wisconsin-Madison, USA
kunapuli@wisc.edu
[3] University of Alberta, Canada
nray1@ualberta.ca
[4] Indiana University, USA
natarasr@indiana.edu

Abstract. We introduce a novel, robust data-driven regularization strategy called Adaptive Regularized Boosting (AR-Boost), motivated by a desire to reduce overfitting. We replace AdaBoost's hard margin with a regularized soft margin that trades-off between a larger margin, at the expense of misclassification errors. Minimizing this regularized exponential loss results in a boosting algorithm that *relaxes the weak learning assumption further*: it can use classifiers with error greater than $\frac{1}{2}$. This enables a natural extension to multiclass boosting, and further reduces overfitting in both the binary and multiclass cases. We derive bounds for training and generalization errors, and relate them to AdaBoost. Finally, we show empirical results on benchmark data that establish the robustness of our approach and improved performance overall.

1 Introduction

Boosting is a popular method for improving the accuracy of a classifier. In particular, AdaBoost [1] is considered the most popular form of boosting and it has been shown to improve the performance of base learners both theoretically and empirically. The key idea behind AdaBoost is that it constructs a strong classifier using a set of weak classifiers [2,3]. While AdaBoost is quite powerful, there are two major limitations: (1) if the base classifier has a misclassification error of greater than 0.5, generalization decreases, and (2) it suffers from overfitting with noisy data [4,5].

The first limitation can become severe in multiclass classification, where the error rate of random guessing is $\frac{C-1}{C}$, where C is the number of classes [6]. AdaBoost requires weak classifiers to achieve an error rate less than 0.5, which can be problematic in multiclass classification. The second limitation of overfitting occurs mainly because weak classifiers are unable to capture "correct" patterns inside noisy data. Noise can be introduced into data by two factors – (1) mislabeled data, or (2) limitation of the hypothesis space of the base classifier [7]. During training, AdaBoost concentrates on learning difficult data patterns accurately,

H. Blockeel et al. (Eds.): ECML PKDD 2013, Part III, LNAI 8190, pp. 1–16, 2013.
© Springer-Verlag Berlin Heidelberg 2013

and potentially distorts the optimal decision boundary. AdaBoost maximizes the "hard margin", namely the smallest margin of those noisy data patterns and consequently *the margin of other data points may decrease significantly.* Different regularization strategies such as early stopping, shrinking the contribution of the individual weak classifiers, and soft margins, have been proposed [2,4,5,7,8,9,10] to combat this issue.

AdaBoost's use of a hard margin increases the penalty exponentially for larger negative margins; this further increases error due to outliers. We propose an approach that combines early convergence with a soft margin by introducing a regularization term inside the exponential loss function. In every boosting round, the regularization term vanishes only if the weak classifier chosen at the current stage classifies the observations correctly. We derive a modified version of the AdaBoost algorithm by minimizing this regularized loss function and this leads to *Adaptive Regularized Boosting* (**AR-Boost**).

We show that choosing optimal values of a data-driven regularized penalty translates to the selection of optimal weights of the misclassified samples at each boosting iteration. These optimal weights force the weak classifiers to correctly label misclassifications in the previous stage. Consequently, AR-Boost *converges faster* than AdaBoost, and is also *more robust* to outliers. Finally, the proposed regularization allows boosting to employ weak classifiers even if their *error rate is greater than* 0.5. This is especially *suited to the multiclass setting*, where the permissible error is $\frac{C-1}{C} > \frac{1}{2}$. This serves as another significant motivation for the development of this approach.

Many properties that motivate this approach are *controlled by the user through tuning* a single regularization parameter $\rho > 1$, and this parameter determines how much differently AR-Boost behaves, compared to AdaBoost. The parameter ρ softens the margin, making our approach more robust to outliers. This is because it does not force classification of outliers according to their (possibly) incorrect labels, and thus does not distort the optimal decision boundary. Instead, it allows *a larger margin at the expense of some misclassification error.* To better understand this, consider the example presented in Figure 1. When the data is noisy, AdaBoost will still aim to classify the noisy example into one of the classes; our approach instead avoids this, leading to a more robust decision boundary. This added robustness allows for better generalization (shown in the bottom row of Figure 1). In addition to an empirical demonstration of this approach's success, we also derive theoretical bounds on the training and generalization error.

The rest of the paper is organized as follows. After reviewing existing work on boosting in Section 2, we describe binary AR-Boost in Section 3, and provide justification for our choice of regularization. In Section 4, we investigate the theoretical properties of our approach by deriving training and generalization error bounds. We describe the multiclass extension of AR-Boost in Section 5. In Section 6, we investigate the empirical properties of binary and multiclass AR-Boost, and compare their performance to some well-known regularized boosting approaches, and conclude in Section 7.

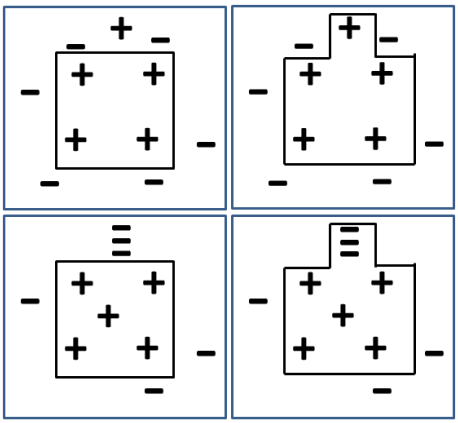

Fig. 1. Decision boundary made by the decision stumps (linear thresholds) used as weak classifiers in AR-Boost (*left column*) and AdaBoost (*right column*) on training (*top row*) and test (*bottom row*) dataset.

2 Background and Related Work

For training data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$, we assume that $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$ for binary classification, and $y_i \in \{1, \ldots, C\}$, for C-class classification. AdaBoost learns a strong classifier $f(\mathbf{x}) = \mathsf{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right)$, by combining weak classifiers in an iterative manner [2]. Here, α_t is the weight associated with the weak classifier $h_t(\cdot)$. The value of α_t is derived by minimizing an exponential loss function: $L(y, f(\mathbf{x})) = \exp(-yf(\mathbf{x}))$.

AdaBoost is prone to overfitting and several strategies were developed to address this issue. Mease and Wyner [4] experimentally demonstrated that boosting often suffers from overfitting run for a large number of rounds. Model selection using Akaike or Bayesian information criteria (AIC/BIC) [11,12] achieved moderate success in addressing overfitting. Hastie *et al.*, [2] proposed ϵ-Boost where they regularize by shrinking the contribution of each weak classifier: $f(\mathbf{x}) = \mathsf{sign}\left(\sum_{t=1}^{T} \nu\alpha_t h_t(\mathbf{x})\right)$. More shrinkage (smaller ν) increases training error over AdaBoost for the same number of rounds, but reduces test error.

Jin *et al.*, [7] proposed Weight-Boost, which uses input-dependent regularization that combines the weak classifier with an instance-dependent weight factor: $f(\mathbf{x}) = \mathsf{sign}\left(\sum_{t=1}^{T} \exp(-|\beta f_{t-1}(\mathbf{x})|)\, \alpha_t h_t(\mathbf{x})\right)$. This trades-off between the weak classifier at the current iteration and the meta-classifier from previous iterations. The factor $\exp(-|\beta f_{t-1}(\mathbf{x})|)$ only considers labels provided by $h_t(\mathbf{x})$ when the previous meta-classifier f_{t-1} is not confident on its decision. Xi *et al.*, [13] minimized an L_1-regularized exponential loss $L(y, f(\mathbf{x})) = \exp(-yf(\mathbf{x}) + \beta\|\alpha\|_1)$, $\beta > 0$, which provides sparse solutions and early stopping. Rätsch *et al.*, [5,9] proposed a weight-decay method, in which they softened the margin by introducing a *slack variable* $\xi = \left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right)^2$ in the exponential loss function

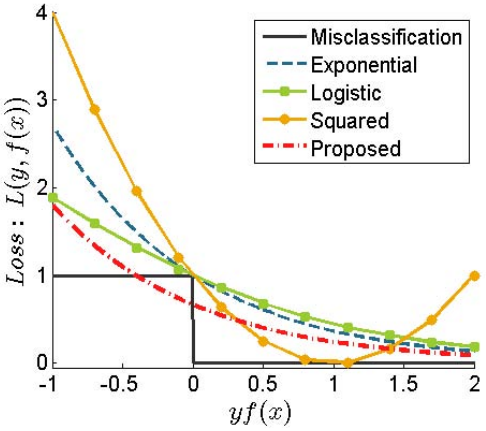

Fig. 2. Common loss functions for binary classification compared with the proposed loss $L(y, f(\mathbf{x}) = \exp(-yf(\mathbf{x}) - \lambda|y - h_t(\mathbf{x})|)$, where h_t is the most recent weak learner.

$L(y, f(\mathbf{x})) = \exp(-yf(\mathbf{x}) - C\xi)$, $C \geq 0$. They found that the asymptotic margin distribution for AdaBoost with noisy data is very similar to that of SVMs [14]; analogous to SVMs, "easy" examples do not contribute to the model and only "difficult" patterns with small margins are useful.

3 AR-Boost for Binary Classification

For any condition π, let $\boldsymbol{\delta}[\![\,\pi\,]\!] = 1$, if π holds, and 0 otherwise. AdaBoost minimizes an exponential loss function, $L(y, f(\mathbf{x})) = \exp(-yf(\mathbf{x}))$. The misclassification loss, $L(y, f(\mathbf{x})) = \boldsymbol{\delta}[\![\,yf(\mathbf{x}) < 0\,]\!]$ penalizes only the misclassified examples (with $yf(\mathbf{x}) < 0$) with an exact penalty of 1. Other loss functions (see Figure 2) attempt to overestimate the discontinuous misclassification loss with continuous/differentiable alternatives. Of these, the squared loss, $L(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$ does not decrease monotonically with increasing margin $yf(\mathbf{x})$. Instead, for $yf(\mathbf{x}) > 0$ it increases quadratically, with increasing influence from observations that are correctly classified with increasing certainty. This significantly reduces the relative influence misclassified examples.

While exponential loss is monotonically decreasing, it penalizes larger misclassified margins exponentially, is exponentially large for these outliers and leads to worse misclassification rates [2]. This motivates the novel loss function,

$$L(y, f(\mathbf{x})) = \exp(-yf(\mathbf{x}) - \lambda|y - h_t(\mathbf{x})|), \tag{1}$$

where $\lambda > 0$ and $h_t(\cdot)$ is the weak classifier chosen at the current step, t[1]. The additional term in the loss function $|y - h_t(\mathbf{x})|$ acts as a regularizer, in

[1] As the loss function incorporates the weak classifier from the last round h_t, it should be written $L_t(y, f(\mathbf{x}))$; we drop the subscript t from L to simplify notation, as the dependence of the loss on t is apparent from the context.

Algorithm 1. AR-Boost for Binary Classification

input: $\lambda = \frac{1}{2}\log\rho$ {select $\rho > 1$ such that $\lambda > 0$}
$w_i^1 = \frac{1}{n}$, $i = 1, \ldots, n$ {initialize example weight distribution uniformly}
for $t = 1$ **do**
  $h_t = \mathsf{WeakLearner}\left((\mathbf{x}_i, y_i, w_i^t)_{i=1}^n \right)$ {train weak classifier h_t using weights w_i^t}
  $\varepsilon_t = \sum_{i=1}^{n} w_i^t \boldsymbol{\delta}[\![y_i \neq h_t(\mathbf{x}_i)]\!]$ {sum of weights of examples misclassified by h_t}
  if $\varepsilon_t \geq \frac{\rho}{\rho+1}$ **then**
    $T = t - 1$; **break**.
  else
    $\alpha_t = \frac{1}{2}\log\frac{\rho(1-\varepsilon_t)}{\varepsilon_t}$ {update α_t with adaptive regularization parameter ρ}
    $w_i^{t+1} = \frac{w_i^t \exp\left(2\alpha_t \boldsymbol{\delta}[\![y_i \neq h_t(\mathbf{x}_i)]\!]\right)}{Z_t}$ {update weights with normalization Z_t}
  end if
end for
output: $f(\mathbf{x}) = \mathsf{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) \right)$ {final classifier}

conjunction with the margin term $yf(\mathbf{x})$. This term does not resemble typical norm-based regularizations that control the structure of the hypothesis space, such as ℓ_1 or ℓ_2-norms. It behaves like a regularization term because, it controls the hypothesis space by relaxing the weak learning assumption in order to admit hypotheses that have error greater than $\frac{1}{2}$ into the boosting process.

At iteration t, the proposed loss function is the same as AdaBoost's loss function if the misclassification error is zero. However, the penalty associated with this loss is less than that of AdaBoost's loss if an example is misclassified (Figure 2). AdaBoost maximizes the hard margin, $\gamma = yf(\mathbf{x})$ without allowing any misclassification error, $\mathsf{e}_{\mathsf{tr}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\delta}[\![y_i f(\mathbf{x}_i) < 0]\!]$. Inspired by SVMs [14], our function maximizes a soft margin, $\gamma = yf(\mathbf{x}) + \lambda|y - h_t(\mathbf{x})|$. Instead of enforcing outliers to be classified correctly, this modification allows for a larger margin *at the expense of some misclassification errors*, $\mathsf{e}'_{\mathsf{tr}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\delta}[\![y_i f(\mathbf{x}_i) + \lambda|y_i - h_t(\mathbf{x}_i)| < 0]\!]$, and tries to avoid overfitting.

We derive a modified AdaBoost algorithm that we call *Adaptive Regularized Boosting* (AR-Boost); the general procedure of AR-Boost for binary classification is shown in Algorithm 1. The derivation of the updates is shown in Appendix A. AR-Boost finds the hypothesis weight, $\alpha_t = \frac{1}{2}\log\rho\frac{(1-\varepsilon_t)}{\varepsilon_t}$, with $\lambda = \frac{1}{2}\log\rho > 0$. When $\rho = 1$, AR-Boost is the same as AdaBoost. As mentioned earlier, the $\mathsf{WeakLearner}$ is capable of learning with classifiers with an error rate $\varepsilon_t > 0.5$. The extent to which this error is tolerated is further discussed in the next section. For all learners, we have $\alpha_t = \lambda + \frac{1}{2}\log\frac{(1-\varepsilon_t)}{\varepsilon_t}$. This is equivalent to computing $\alpha_t^{\mathsf{AR\text{-}Boost}} = \lambda + \alpha_t^{\mathsf{AdaBoost}}$.

One additional advantage of this regularized loss function is that the penalty for negative margins can be adjusted after observing the classifier performance. Accordingly, we determine the value of λ or ρ through cross validation, by choosing the parameter for which the average misclassification error, $\mathsf{e}'_{\mathsf{tr}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\delta}[\![-y_i f(\mathbf{x}_i) - \lambda|y_i - h_t(\mathbf{x}_i)| < 0]\!]$ is smallest. For instance, in Figure 3

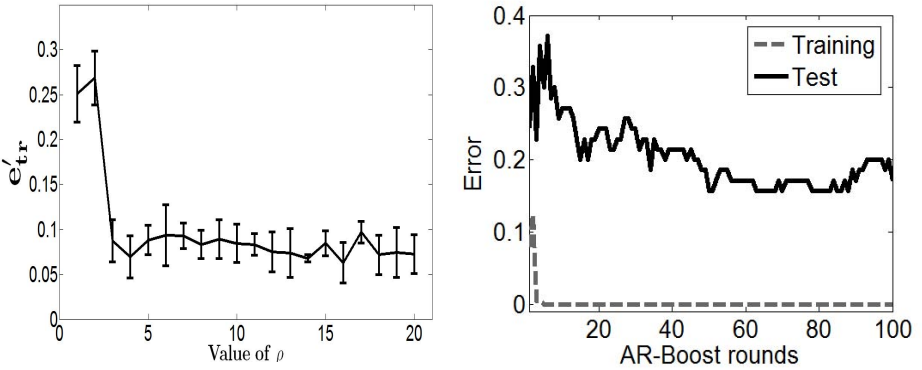

Fig. 3. PIMA Indian Diabetes data set (**left**) 5-fold cross validation with smallest cross-validation error for $\rho = 4$; (**right**) training and test errors over a number of boosting iterations for ρ=4

(left), we show AR-Boost's 5-fold cross-validation error for the PIMA Indian Diabetes data set [7]. The best value is $\rho = 4$, and the corresponding training and test error curves are shown in Figure 3 (right) for this choice of ρ. This behavior is similar to AdaBoost, in that even when the training error has been minimized, the test error continues to decrease.

We derive the multiclass version of AR-Boost in Section 5, which takes advantage of AR-Boost's ability to handle weaker classifiers. Before proceeding, we further analyze AR-Boost's ability to relax the weak-learner assumption.

3.1 Relaxing the Weak Learning Assumption of AdaBoost

At the t-th iteration, $\alpha_t = \frac{1}{2}\log\frac{\rho(1-\varepsilon_t)}{\varepsilon_t}$, $\rho > 1$. The hypothesis weight $\alpha_t > 0$ only when $\frac{\rho(1-\varepsilon_t)}{\varepsilon_t} > 1$. From this, it is immediately apparent that

$$\varepsilon_t < \frac{\rho}{1+\rho}, \tag{2}$$

and when $\rho = 1$, we have that $\varepsilon_t < 0.5$; this is the standard weak learning assumption that is used in AdaBoost. As we start increasing the value of $\rho > 1$, we can see that $\frac{\rho}{\rho+1} \to 1$ and AR-Boost is able to accommodate classifiers with $\varepsilon_t \in [0.5, \frac{\rho}{\rho+1})$. Thus, AR-Boost is able to *learn with weaker hypotheses than afforded by the standard weak learning assumption*; how weak these learners can be is controlled by the choice of ρ. This can be seen in Figure 4, which shows the AR-Boost objective values, and their minima plotted for various weak-learner errors ε. AR-Boost can handle weaker classifiers than AdaBoost, and assigns them increasingly lower weights $\alpha \to 0$, the weaker they are.

3.2 How Does Relaxing the Weak Learning Assumption Help?

Similar to Zhu *et al.*, [6] who illustrate that AdaBoost fails in the multiclass setting, we show how this can also happen in binary classification. We conduct

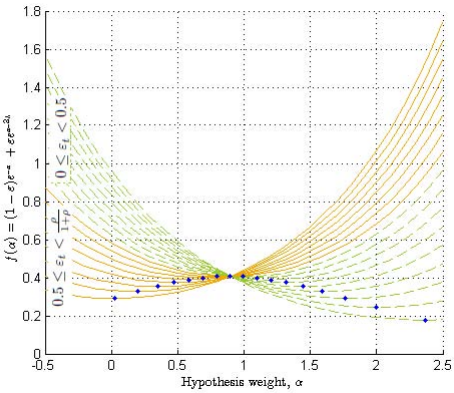

Fig. 4. Objective values of AR-Boost ($f(\alpha) = (1-\varepsilon)e^{-\alpha} + \varepsilon e^{\alpha-2\lambda}$) plotted as a function of α for different values of ε. The minimum of each objective is also shown by dot. The curves in dashed green represent classifiers that satisfy the weak learner assumption: $0 \le \varepsilon < 0.5$, while the curves in orange represent classifiers that exceed it: $0.5 \le \varepsilon < \frac{\rho}{\rho+1}$. These curves were plotted for $\rho = 6$.

an experiment with simple two-class data, where each example $\mathbf{x} \in \mathbb{R}^{10}$, and $x_{ij} \sim \mathcal{N}(0,1)$. The two classes are defined as,

$$c = \begin{cases} 1, & \text{if } 0 \le \sum x_j^2 < \chi^2_{10,1/2}, \\ -1 & \text{otherwise,} \end{cases}$$

where $\chi^2_{10,1/2}$ is the $(1/2)100\%$ quantile of the χ^2_{10} distribution. The training and test set sizes were 2000 and $10,000$, with class sizes being approximately equal. We use decision stumps (single node decision tree) as weak learners.

Figure 5 (top row) demonstrates how AdaBoost sometimes fails in binary classification. Training and test errors remain unchanged over boosting rounds (Figure 5 top left). The error ε_t and the AdaBoost weights α_t for each round t are shown in Figure 5 top center, and right. The value of ε_t starts below $\frac{1}{2}$, and after a few iterations, it overshoots $\frac{1}{2}$ ($\alpha_t < 0$), then is quickly pushed back down to $\frac{1}{2}$ (Figure 5 top center). Now, once ε_t is equal to $\frac{1}{2}$, the weights of subsequent examples *are no longer updated* ($\alpha_t = 0$). Thus, no new classifiers are added to $f(\mathbf{x})$, and the *overall error rate remains unchanged*.

Unlike AdaBoost, AR-Boost *relaxes the weak learning assumption further*: it can use classifiers with error greater than $\frac{1}{2}$ as shown in bottom row of Figure 5. Here, both training and test error decrease with boosting iterations, which is what we would expect to see from a successful boosting algorithm. AR-Boost can successfully incorporate weak classifiers with error as large as $\varepsilon_t < \rho/(\rho+1)$ for binary classification as shown in Algorithm 1. Similar behavior holds for the C-class case, which can incorporate classifiers with error up to $\varepsilon < \rho(C-1)/(\rho(C-1)+1)$ as we show below, in Algorithm 2. This limiting value of ε_t is not artificial, it follows naturally by minimizing the proposed novel regularized

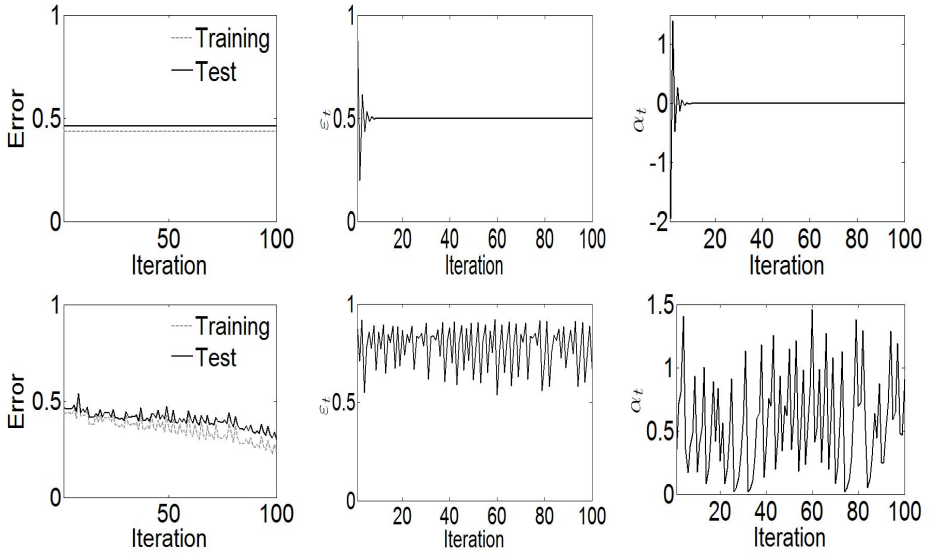

Fig. 5. Comparing AdaBoost and AR-Boost on a simple two-class simulated example, with decision stumps (single node decision trees) used as weak learners. The top row (AdaBoost) and bottom row (AR-Boost) show (**left**) training and test errors; (**center**) weak classifier error at round t, ε_t; and (**right**) example weight at round t, α_t respectively for AdaBoost and AR-Boost. For AR-Boost, $\rho = 5$.

exponential loss function. It provides softer margins and smaller penalties for larger negative margins than AdaBoost (Figure 2).

4 Analysis of AR-Boost

We now analyze the behavior of AR-Boost via upper bounds on the training and generalization error and compare these bounds with AdaBoost.

4.1 Training Error

We can formally analyze the behavior of the algorithm by deriving an upper bound on the training error. To do so, we first state the following.

Lemma 1. *At the t-th iteration, define the* goodness γ_t *of the current weak learner* $h_t(\mathbf{x})$ *as how much better it does than the worst allowable error:* $\varepsilon_t = \frac{\rho}{\rho+1} - \gamma_t$. *The normalization* Z_t *of the weights* w_i^{t+1} *(in Algorithm 1) can be bounded by*

$$Z_t \leq \exp\left(-\frac{(\rho+1)^2}{2\rho}\gamma_t^2 + \frac{\rho^2-1}{2\rho}\gamma_t\right). \tag{3}$$

This Lemma is proved in Appendix C. Now, we can state the theorem formally.

Theorem 1. *If the goodness of weak learners at every iteration is bounded by* $\gamma_t \geq \gamma$*, the training error of AR-Boost,* e_{tr}*, after* T *rounds is bounded by*

$$\mathsf{e}_{\mathsf{tr}} \leq \prod_{t=1}^{T} \exp\left(-\frac{(\rho+1)^2}{2\rho}\gamma^2 + \frac{\rho^2-1}{2\rho}\gamma\right). \tag{4}$$

If $\gamma \geq \frac{\rho-1}{\rho+1}$*, the training error exponentially decreases.*

Proof. After T iterations, the example weights w_i^{T+1} can be computed using step 9 of Algorithm 1. By recursively unraveling this step, and recalling that $w_i^1 = \frac{1}{n}$, we have

$$w_i^{T+1} = \frac{e^{-y_i f(\mathbf{x}_i)}}{n \prod_{t=1}^{T} Z_t}. \tag{5}$$

The training error is $\mathsf{e}_{\mathsf{tr}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\delta}[\![\, y_i \neq f(\mathbf{x}_i)\,]\!]$. For all misclassified examples, we can bound the training error by

$$\mathsf{e}_{\mathsf{tr}} \leq \frac{1}{n}\sum_{i=1}^{n} e^{-y_i f(\mathbf{x}_i)} = \prod_{t=1}^{T} Z_t,$$

where we use (5) and the fact that $\sum_{i=1}^{n} w_i^{t+1} = 1$. The bound follows from Lemma 1 and the fact that $\gamma_t \geq \gamma$. □

First, note that when $\rho = 1$, the training error bound is exactly the same as that of AdaBoost. Next, to understand the behavior of this upper bound, consider Figure 6. The bound of AdaBoost is shown as the dotted line, while the remaining curves are the AR-Boost training error for various values of $\rho > 1$. It is evident that it is possible to *exponentially shrink the training error* for increasing T, as long as the *goodness of the weak learners* is at least $\gamma = \frac{\rho-1}{\rho+1}$, which means that the error at each iteration, $\varepsilon_t \leq \frac{1}{\rho+1}$.

4.2 Generalization Error

Given a distribution D over $X \times \{\pm 1\}$ and a training sample S drawn i.i.d. from D, Schapire *et al.*, [15] showed that the upper bound on the generalization error of AdaBoost is, with probability $1-\delta$, $\forall\theta > 0$,

$$\mathsf{Pr}_{\mathsf{D}}[\![\, yf(\mathbf{x}) \leq 0\,]\!] \leq \mathsf{Pr}_{\mathsf{S}}[\![\, yf(\mathbf{x}) \leq \theta\,]\!] + O\left(\frac{1}{\sqrt{n}}\left(\frac{d\log^2(\frac{n}{d})}{\theta^2} + \log\frac{1}{\delta}\right)^{\frac{1}{2}}\right), \tag{6}$$

where d is the Vapnik-Chervonenkis (VC) dimension of the space of base classifiers. This bound depends on the training error $\mathsf{e}_{\mathsf{tr}}^{\theta} = \mathsf{Pr}_{\mathsf{S}}[\![\, yf(\mathbf{x}) \leq \theta\,]\!]$ and is *independent* of the number of boosting rounds T.

Schapire *et al.*, explained AdaBoost's ability to avoid overfitting using the margin, $m = yf(\mathbf{x})$, the magnitude of which represents the measure of confidence

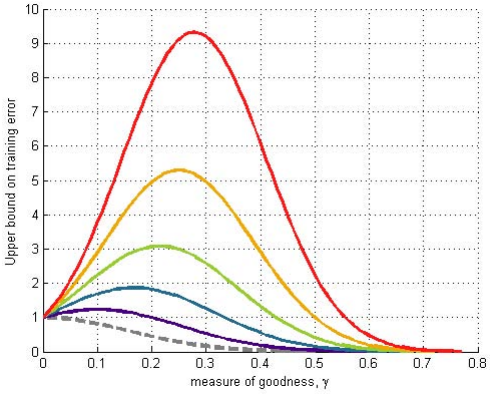

Fig. 6. The upper bound on the training error for various choices of ρ, as the goodness of the weak learners improves. AdaBoost ($\rho = 1$) is the dashed line. The behavior of AR-Boost for various values of $\rho > 1$ is similar to AdaBoost if $\gamma > \frac{\rho-1}{\rho+1}$, that is, the training error decreases exponentially as $T \to \infty$.

on the predictions of the base classifiers. The bound above shows that a large margin on the training set results in a superior bound on the generalization error. Since boosting minimizes the exponential loss $L(y, f(\mathbf{x})) = e^{-yf(\mathbf{x})}$, the margin is maximized. The result below shows that AR-Boost also behaves similarly to maximize the margin, $m = yf(\mathbf{x}) + \lambda|y - h_t(\mathbf{x})|$, where $\lambda > 0$.

Theorem 2. *At every iteration $t = 1, \ldots, T$, let the base learner produce classifiers with training errors ε_t. Then, for any $\theta > 0$, we have*

$$\mathsf{e}_{\mathsf{tr}}^{\theta} \;=\; \mathsf{Pr}_{\mathsf{S}}[\![\, yf(\mathbf{x}) \le \theta \,]\!] \;\le\; \left(\sqrt{\rho^{1+\theta}} + \frac{1}{\sqrt{\rho^{1-\theta}}}\right)^{T} \prod_{t=1}^{T} \sqrt{\varepsilon_t^{1-\theta}(1-\varepsilon_t)^{1+\theta}}. \quad (7)$$

Proof. If $yf(\mathbf{x}) \le \theta$, then $y\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) \le \theta \sum_{t=1}^{T} \alpha_t$ and $\exp(-y\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) + \theta \sum_{t=1}^{T} \alpha_t) \ge 1$. Using this, we have that

$$\begin{aligned} \mathsf{Pr}_{\mathsf{S}}[\![\, yf(\mathbf{x}) \le \theta \,]\!] &\le \mathsf{E}_{\mathsf{S}}\left[\!\!\left[\exp\left(-y\textstyle\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) + \theta \sum_{t=1}^{T} \alpha_t\right) \right]\!\!\right] \\ &= \frac{1}{n} \exp\left(\theta \sum_{t=1}^{T} \alpha_t\right) \sum_{i=1}^{n} \exp\left(-y_i f(\mathbf{x}_i)\right). \end{aligned}$$

Using the value of α_t, and equations (5) and (10) gives us the result. □

As before, we immediately note that when $\rho = 1$, this bound is exactly identical to the bound derived by Schapire *et al.* [15]. To further analyze this bound, assume that we are able to produce classifiers with $\varepsilon_t \le \frac{\rho}{\rho+1} - \gamma$, with some goodness $\gamma > 0$. We know from Theorem 1 that the upper bound of training reduces exponentially with T. Then, we can simplify the upper bound in (7) to

$$\mathsf{Pr}_{\mathsf{S}}[\![\, yf(\mathbf{x}) \le \theta \,]\!] \le \left(\frac{1}{\rho} - \frac{(\rho+1)}{\rho}\gamma\right)^{\frac{(1-\theta)T}{2}} (1 + (\rho+1)\gamma)^{\frac{(1+\theta)T}{2}}$$

Algorithm 2. AR-Boost for multiclass Classification

input: $\lambda = \frac{C-1}{2} \log \rho$ {select $\rho > 1$ such that $\lambda > 0$}
$w_i^1 = \frac{1}{n}$, $i = 1, \ldots, n$ {initialize example weight distribution uniformly}
for $t = 1$ **do**
  $H_t = \mathsf{WeakLearner}\left((\mathbf{x}_i, \mathbf{y}_i, w_i^t)_{i=1}^n \right)$ {train weak classifier H_t using weights w_i^t}
  $\varepsilon_t = \sum_{i=1}^{n} w_i^t \boldsymbol{\delta}[\![c_i \neq H_t(\mathbf{x}_i)]\!]$ {sum of weights of examples misclassified by H_t}
  if $\varepsilon_t \geq \frac{\rho(C-1)}{\rho(C-1)+1}$ **then**
    $T = t - 1$; **break**.
  else
    $\alpha_t = \log \frac{\rho(1-\varepsilon_t)}{\varepsilon_t} + \log(C-1)$ {update α_t with adaptive parameter ρ}
    $w_i^{t+1} = \frac{w_i^t \exp(\alpha_t \boldsymbol{\delta}[\![c_i \neq H_t(\mathbf{x}_i)]\!])}{Z_t}$ {update weights with normalization Z_t}
  end if
end for
output: $f(\mathbf{x}) = \arg\max_k \left(\sum_{t=1}^{T} \alpha_t \boldsymbol{\delta}[\![H_t(\mathbf{x}) = k]\!] \right)$ {final classifier}

Hence, $\forall \theta < \frac{1}{2}\frac{\rho-1}{\rho+1}$, and $\forall \gamma > \frac{\rho-1}{\rho+1}$, we have that $\mathsf{Pr}_{\mathsf{S}}[\![yf(\mathbf{x}) \leq \theta]\!] \to 0$ as $T \to \infty$. This suggests that $\lim_{T\to\infty} \min_i y_i f(\mathbf{x}_i) \geq \gamma$, showing that better weak hypotheses, with greater γ, provide larger margins.

5 AR-Boost for Multiclass Classification

In the C-class classification case, each data point can belong to one of C classes i.e., the label of the i-th data point $c_i \in 1, \ldots, C$. For this setting, we can recode the output as a C-dimensional vector $\mathbf{y}_i$ [6,16] whose entries are such that $y_i^k = 1$, if $c_i = k$; else $y_i^k = -\frac{1}{C-1}$, if $c_i \neq k$. The set of C possible output vectors for a C-class problem is denoted by Υ. Given the training data, we wish to find a C-dimensional vector function $\mathbf{f}(\mathbf{x}) = \left(f^1(\mathbf{x}), ..., f^C(\mathbf{x})\right)'$ such that

$$\mathbf{f}(\mathbf{x}) = \begin{array}{c} \arg\min_{\mathbf{f}} \ \sum_{i=1}^{n} L(\mathbf{y}_i, \mathbf{f}(\mathbf{x}_i)) \\ \text{subject to } \sum_{k=1}^{C} f^k(\mathbf{x}) = 0. \end{array}$$

We consider $\mathbf{f}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \mathbf{h}_t(\mathbf{x})$, where $\alpha_t \in \mathbb{R}$ are coefficients, and $\mathbf{h}_t(\mathbf{x})$ are basis functions. These functions $\mathbf{h}_t(\mathbf{x}) : X \to \Upsilon$ are required to satisfy the symmetric constraint: $\sum_{k=1}^{C} h_t^k(\mathbf{x}) = 0$. Finally, every $\mathbf{h}_t(\mathbf{x})$ is associated with a multiclass classifier $H_t(\mathbf{x})$ as, $h_t^k(\mathbf{x}) = 1$, if $H_t(\mathbf{x}) = k$; else $h_t^k(\mathbf{x}) = -\frac{1}{C-1}$, if $H_t(\mathbf{x}) \neq k$, such that solving for $\mathbf{h}_t$ is equivalent to finding the multiclass classifier $H_t : X \to \{1, \ldots, C\}$; in turn, $H_t(\mathbf{x})$ can generate $h_t(\mathbf{x})$ resulting in a one-to-one correspondence.

The proposed multiclass loss function for AR-Boost is $L(\mathbf{y}, \mathbf{f}(\mathbf{x})) = \exp\left(-\frac{1}{C}\mathbf{y}'\mathbf{f}(\mathbf{x}) - \frac{\lambda}{C}\|\mathbf{y} - \mathbf{h}_t(\mathbf{x})\|_1\right)$, which extends the binary loss function to the multiclass case discussed in Section 3. This loss is the natural generalization of the exponential loss for binary classification proposed by Zhu *et al.*, [6] as

Table 1. Description of binary and multiclass data sets used in the experiments

Dataset	#examples	#Train	#Test	#features	#classes
ionosphere	351	238	113	341	2
german	1000	675	325	20	2
diabetes	768	531	237	8	2
wpbc	198	141	57	30	2
wdbc	569	423	146	30	2
spambase	4601	3215	1386	58	2
vowel	990	528	462	13	11
pen digits	10992	7494	3498	16	10
letter	20000	16000	4000	16	26
thyroid	215	160	55	5	3
satimage	6435	4435	2000	36	7
segmentation	2310	210	2100	19	7

Stage-wise Additive Modeling that uses a multiclass Exponential loss function (SAMME). The general procedure of multiclass AR-Boost is shown in Algorithm 2 and the details of the derivation are shown in Appendix B. AR-Boost finds the feature weight, $\alpha_t = \log \rho \frac{1-\varepsilon_t}{\varepsilon_t} + \log(C-1)$, with $\rho > 1$. When $\rho = 1$, the AR-Boost algorithm becomes the SAMME algorithm.

6 Experimental Results

We compare AR-Boost with AdaBoost and four other regularized boosting algorithms: ϵ-boost [2], L_1-regularized boost [13], AdaBoost$_{reg}$ [5] and WeightBoost [7]. We chose 12 data sets (6 binary and 6 multiclass problems) (see Table 1) from the UCI machine learning repository [17] that have been previously used in literature [6,7]. For all the algorithms, the maximum training iterations is set to 100. We also use classification and regression trees (CART) [2] as the baseline algorithm. We compared multiclass AR-Boost discussed in Section 5 to two commonly used algorithms: AdaBoost.MH [18] and SAMME [6]. The parameter ρ was tuned through cross validation.

Figure 7 (left) shows the results of binary classification across 6 binary classification tasks. The baseline decision tree has the worst performance and AdaBoost improves upon trees. Using regularization, however, gives different levels of improvement over AdaBoost. Our AR-Boost approach yields further improvements compared to the other regularized boosting methods on all data sets except spambase. On spambase, AR-Boost produces test error of 4.91%, while Weight Decay and WeightBoost give errors of 4.5% and 4.2% respectively. These results demonstrate that performance is significantly improved for smaller data sets (for example, the improvement is nearly 35% for wpbc). This shows that AR-Boost is able to reduce overfitting, significantly at times, and achieves better generalization compared to state-of-the-art regularized boosting methods on binary classification problems.

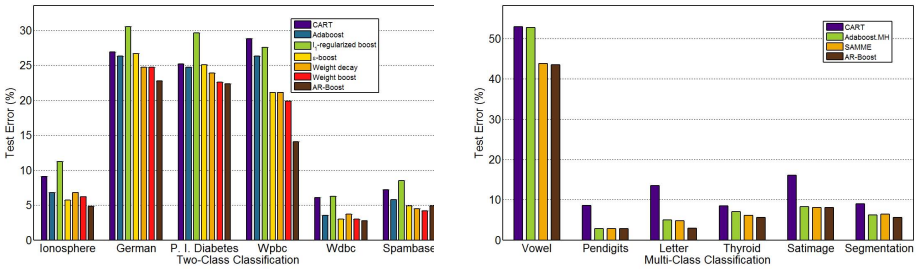

Fig. 7. Misclassification errors of binary AR-Boost (**left**) and multiclass AR-Boost (**right**) compared with other AdaBoost algorithms on UCI data sets (see Table 1).

Similar results are observed in the multiclass setting (Figure 7, right). Boosting approaches are generally an order of magnitude better than the baseline; AR-Boost is comparable (in 3 tasks) or better (in 3 tasks) than SAMME, the best of the other methods. The most interesting result is found on the `vowel` dataset; both AR-Boost and SAMME achieve around 40% test error, which is almost 15% better than AdaBoost.MH. This demonstrates that our approach can seamlessly extend to the multiclass case as well. Again, similar to the binary case, AR-Boost improves robustness to overfitting, especially for smaller data sets (for example, nearly 33% improvement for the `thyroid` data set).

Finally, we investigate an important property of AR-Boost: robustness to outliers, which is a prime motivation of this approach. In this experiment, we introduced different levels of label noise (10%, 20%, 30%) in the binary classification tasks, and compared AR-Boost to the baseline and AdaBoost. We randomly flip the label to the opposite class for random training examples for the benchmark data. Increasing levels of noise: 10%, 20% and 30% were introduced, with those probabilities of flipping a label. AR-Boost exhibits superior performance (Figure 8) at all noise levels. The key result that needs to be emphasized is that at higher noise levels, the difference becomes more pronounced. This suggests that AR-Boost is reasonably *robust* to increasing noise levels, while performance decreases for other approaches, sometimes drastically. Thus, AR-Boost can learn successfully in various noisy settings, and also with small data sets.

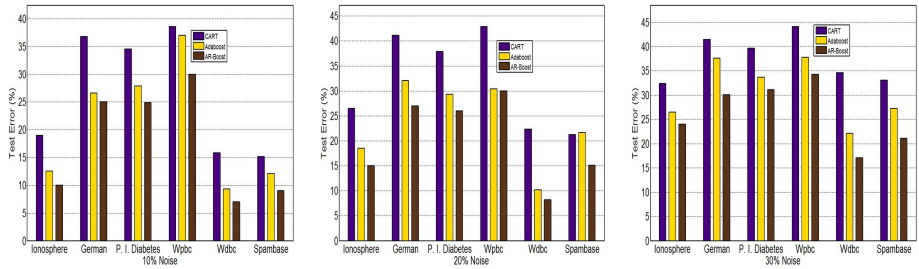

Fig. 8. Misclassification errors of CART, AdaBoost and AR-Boost on UCI datasets with 10%, 20% ad 30% label noise.

These results taken together demonstrate that AR-Boost addresses two limitations in AdaBoost, and other regularized boosting approaches to date, which are also two motivating objectives: robustness to noise, and ability to effectively handle multiclass classification.

7 Conclusion

We proposed Adaptive Regularized Boosting (AR-Boost) that appends a regularization term to AdaBoost's exponential loss. This is a data-driven regularization method, which softens the hard margin of AdaBoost by assigning a smaller penalty to misclassified observations at each boosting round. Instead of forcing outliers to be labelled correctly, AR-Boost allows a larger margin; while this comes at the cost of some misclassification errors, it improves robustness to noise. Compared to other regularized AdaBoost algorithms, AR-Boost uses weaker classifiers, and thus can be used in the multiclass setting. The upper bound of training and generalization error of AR-Boost illustrate that the error rate decreases exponentially with boosting rounds. Extensive experimental results show that AR-Boost outperforms state-of-the-art regularized AdaBoost algorithms for both binary and multiclass classification. It remains an interesting future direction to understand the use of such an approach in other problems such as semi-supervised learning and active learning.

Acknowledgements. Sriraam Natarajan (SN) gratefully acknowledges the support of Defense Advanced Research Projects Agency (DARPA) DEFT Program under Air Force Research Laboratory (AFRL) prime contract FA8750-13-2-0039. SN and Gautam Kunapuli gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract FA8750-09-C-0181. Any opinions, findings, and conclusion expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

References

1. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on- line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139 (1997)
2. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edn. Springer (2009)
3. Schapire, R.E., Freund, Y.: Boosting Foundations and Algorithms, 1st edn. MIT Press (2012)
4. Mease, D., Wyner, A.: Evidence contrary to the statistical view of boosting. Journal of Machine Learning Research, 131–156 (1998)
5. Rätsch, G., Onoda, T.: Müller, K.R.: An improvement of AdaBoost to avoid overfitting. In: Proc. ICONIP, pp. 506–509 (1998)
6. Zhu, J., Zhou, H., Rosset, S., Hastie, T.: Multi-class AdaBoost. Statistics and Its Inference 2, 349–360 (2009)

7. Jin, R., Liu, Y., Si, L., Carbonell, J., Hauptmann, A.G.: A new boosting algorithm using input-dependent regularizer. In: Proc. ICML, pp. 615–622 (2003)
8. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. The Annals of Statistics 28, 2000 (1998)
9. Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for AdaBoost. In: Machine Learning, pp. 287–320 (2000)
10. Sun, Y., Li, J., Hager, W.: Two new regularized AdaBoost algorithms. In: Proc. ICMLA (2004)
11. Bühlmann, P., Hothorn, T.: Boosting algorithms: Regularization, prediction and model fitting. Statistical Science, 131–156 (2007)
12. Kaji, D., Watanabe, S.: Model selection method for adaBoost using formal information criteria. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008, Part II. LNCS, vol. 5507, pp. 903–910. Springer, Heidelberg (2009)
13. Xi, Y.T., Xiang, Z.J., Ramadge, P.J., Schapire, R.E.: Speed and sparsity of regularized boosting. In: Proc. AISTATS, pp. 615–622 (2009)
14. Vapnik, V.: The Nature of Statistical Learning Theory. Springer (2000)
15. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods (1998)
16. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. Journal of the American Statistical Association 99, 67–81 (2004)
17. Bache, K., Lichman, M.: UCI Machine Learning Repository (2013), `http://archive.ics.uci.edu/ml`
18. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. In: Machine Learning, vol. 37, pp. 297–336 (1999)

Appendix

A Derivation of Binary AR-Boost

At the t-th iteration, the loss function is $L(y, f(\mathbf{x})) = \exp\left(-yf(\mathbf{x}) - \lambda|y - h_t(\mathbf{x})|\right)$. Let $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + \alpha_t h_t(\mathbf{x})$ be the strong classifier composed of first t classifiers. We have that $\alpha_t = \arg\min_\alpha \sum_{i=1}^n \exp\left(-y_i f_{t-1}(x_i) - \alpha y_i h_t(\mathbf{x}_i) - \lambda|y_i - h_t(\mathbf{x}_i)|\right)$. Using the fact that $w_i^t = \exp\left(-y_i f_{t-1}(x_i)\right)$, we have

$$\begin{aligned}\alpha_t &= \arg\min_\alpha \sum_{i=1}^n w_i^t \exp\left(-\alpha y_i h_t(\mathbf{x}_i) - \lambda|y_i - h_t(\mathbf{x}_i)|\right) \\ &= \arg\min_\alpha \sum_{i:y_i=h_t(\mathbf{x}_i)} w_i^t e^{-\alpha} + \sum_{i:y_i\neq h_t(\mathbf{x}_i)} w_i^t e^{\alpha-2\lambda}.\end{aligned}$$

Letting $\varepsilon_t = \sum_{i=1}^n w_i^t \boldsymbol{\delta}[\![\, y_i \neq h_t(\mathbf{x}_i)\,]\!]$, and observing that w_i^t are normalized ($\sum_{i=1}^n w_i^t = 1$), we have $\alpha_t = \arg\min_\alpha e^{-\alpha}(1-\varepsilon_t) + e^{\alpha-2\lambda}\varepsilon_t$. This gives us

$$\alpha_t = \frac{1}{2}\log\frac{\rho(1-\varepsilon_t)}{\varepsilon_t}, \tag{8}$$

where we use $\lambda = \frac{1}{2}\log\rho$. Now, observing from above that $w_i^{t+1} = \exp\left(-y_i f_t(x_i)\right) = w_i^t \exp\left(-y_i\alpha_t h_t(\mathbf{x}_i)\right)$ and using the fact that $-y_i h_t(\mathbf{x}_i) = 2\boldsymbol{\delta}[\![\, y_i \neq h_t(\mathbf{x}_i)\,]\!] - 1$, we have

$$w_i^{t+1} = w_i^t \exp\left(2\alpha_t \boldsymbol{\delta}[\![\, y_i \neq h_t(\mathbf{x}_i)\,]\!]\right). \tag{9}$$

The term $e^{-\alpha_t}$ is dropped as it appears in $w_i^{t+1}\ \forall i$ and cancels during normalization. Then w_i^{t+1} is expressed in terms of w_i^t, y, α_t and h_t. Subsequently, the summation breaks into two parts: $y = h_t$ and $y \neq h_t$ and finally it uses (8), to get the final expression for Z_t:

$$Z_t \;=\; \sum_{i=1}^{n} w_i^{t+1} \;=\; \left(\frac{1}{\sqrt{\rho}} + \sqrt{\rho}\right)\, \sqrt{\varepsilon_t\,(1-\varepsilon_t)}. \tag{10}$$

B Derivation of Multiclass AR-Boost

At the t-th iteration, the loss function is $L(\mathbf{y}, \mathbf{f}(\mathbf{x})) = \exp\left(-\frac{1}{C}\mathbf{y}'\mathbf{f}(\mathbf{x}) - \frac{\lambda}{C}\|\mathbf{y} - \mathbf{h}_t(\mathbf{x})\|_1\right)$. Let $\mathbf{f}_t(\mathbf{x}) = \mathbf{f}_{t-1}(\mathbf{x}) + \alpha_t\mathbf{h}_t(\mathbf{x})$ be the strong classifier composed of first t classifiers. We need $\alpha_t = \arg\min_\alpha \sum_{i=1}^n \exp\left(-\frac{1}{C}\mathbf{y}_i'(\mathbf{f}_{t-1}(\mathbf{x}_i) + \alpha_t\mathbf{h}_t(\mathbf{x}_i)) + \frac{\lambda}{C}\|\mathbf{y}_i - \mathbf{h}_t(\mathbf{x}_i)\|_1\right)$. Analogous to the two class case, we have $w_i^t = \exp\left(-\frac{1}{C}\mathbf{y}_i'\mathbf{f}_{t-1}(\mathbf{x}_i)\right)$. Recall, that solving for $\mathbf{h}_t(\mathbf{x})$ is encoded as finding the multiclass classifier $H_t(\mathbf{x})$ that yields $\mathbf{h}_t(\mathbf{x})$. Thus, we have

$$\begin{aligned}\alpha_t &= \arg\min_\alpha \sum_{i=1}^{n} w_i^t \exp\left(-\frac{\alpha}{C}\mathbf{y}_i'\mathbf{h}_t(\mathbf{x}_i) - \frac{\lambda}{C}\|\mathbf{y}_i - h_t(\mathbf{x}_i)\|_1\right)\\ &= \arg\min_\alpha \sum_{i:c_i=H_t(\mathbf{x}_i)} w_i^t e^{-\frac{\alpha}{C-1}} + \sum_{i:c_i\neq H_t(\mathbf{x}_i)} w_i^t e^{\frac{\alpha}{(C-1)^2} - \frac{2\lambda}{C-1}}.\end{aligned}$$

As before, we set $\varepsilon_t = \sum_{i=1}^n w_i^t \boldsymbol{\delta}[\![\, c_i \neq H_t(\mathbf{x}_i)\,]\!]$ and we get $\hat{\alpha}_t = \frac{(C-1)^2}{C}\alpha_t$

$$\alpha_t \;=\; \log\frac{\rho(1-\varepsilon_t)}{\varepsilon_t} \;+\; \log(C-1), \tag{11}$$

where $\lambda = \frac{C-1}{2}\log\rho$. This allows us to write

$$w_i^{t+1} \;=\; \begin{cases} w_i^t e^{-\frac{C-1}{C}\alpha_t}, & \text{if } c_i = H_t(\mathbf{x}_i),\\ w_i^t e^{\frac{1}{C}\alpha_t} \quad\;, & \text{if } c_i \neq H_t(\mathbf{x}_i).\end{cases} \tag{12}$$

After normalization, this weight above is equivalent to the weight used in Algorithm 2. Finally, it is simple to show that the output after T iterations, $\mathbf{f}_T(\mathbf{x}) = \arg\max_k(f_T^1(\mathbf{x}), \ldots, f_T^k(\mathbf{x}), \ldots, f_T^C(\mathbf{x}))'$ and is equivalent to $\mathbf{f}_T(\mathbf{x}) = \arg\max_k \sum_{t=1}^T \alpha_t \boldsymbol{\delta}[\![\, H_t(\mathbf{x}) = k\,]\!]$.

C Proof for Lemma 1

At the t-th iteration, setting $\varepsilon_t = \frac{\rho}{\rho+1} - \gamma_t$ in (10) and simplifying gives us

$$Z_t = \frac{\rho+1}{\sqrt{\rho}}\sqrt{\left(\frac{\rho}{\rho+1} - \gamma_t\right)\left(\frac{1}{\rho+1} + \gamma_t\right)} = \sqrt{1 - \frac{(\rho+1)^2}{\rho}\gamma_t^2 + \frac{\rho^2-1}{\rho}\gamma_t}$$

Now, using $1 + x \leq e^x$ gives (3). □