Iterative Weak Learnability and Multiclass AdaBoost

In-Koo Cho
icho30@emory.edu
Emory University
Atlanta, GA
Hanyang University
Seoul, Korea

Jonathan A. Libgober
libgober@usc.edu
University of Southern California
Los Angeles, CA

Cheng Ding
cheng.ding.emoryecon@gmail.com
Emory University
Atlanta, GA

ABSTRACT

We propose an efficient boosting algorithm for multiclass classification, called AdaBoost.Iter, that extends SAMME [14] and AdaBoost [4]. The algorithm iteratively applies the weak learnability condition of SAMME to eliminate classes to find the correct classificiation. The iterative weak learnability is a sufficient and necessary condition for boostability, but it is also easier to validate than the EOR criterion of AdaBoost.MM [9]. We show that the training error of AdaBoost.Iter vanishes at the exponential rate, while the generalization error converges to zero at the same rate as AdaBoost. AdaBoost.Iter numerically outperforms SAMME and achieves performance comparable to AdaBoost.MM on benchmark datasets.

CCS CONCEPTS

• **Theory of computation → Boosting**.

KEYWORDS

Adaptive Boosting Algorithm, Iterative Weak Learnability, Boostability, Multiclass Classification

ACM Reference Format:
In-Koo Cho, Jonathan A. Libgober, and Cheng Ding. 2024. Iterative Weak Learnability and Multiclass AdaBoost. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671842

1 INTRODUCTION

AdaBoost [4] stands as a cornerstone in ensemble algorithm for binary classification, generating highly accurate classifiers efficiently by aggregating multiple weak classifiers. A multiclass extension of AdaBoost is critical for addressing a wider array of applications that involve multiple classes. Inspired by [5], [14] developed a multiclass extension of AdaBoost [4] called SAMME (Stagewise Additive Modeling using a Multiclass Exponential loss function). SAMME maintains the core structure of AdaBoost, applying the same criterion for weak learnability as AdaBoost requiring that classifiers slightly outperform random guesses.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671842

[9] demonstrated that the weak learnability criterion of SAMME does not guarantee boostability in multiclass scenarios, failing to guarantee that an ensemble of weak learners generates a strong classifier in SAMME. [9] proposed Edge over Random (EOR) as a stronger notion of weak learnability to recover the boostability and developed a multiclass extension known as AdaBoost.MM.

EOR uses a non-linear scoring function to evaluate a weak learner. In contrast, the scoring function for the original weak learnability criterion of AdaBoost [4] and SAMME [14] is a linear function of the performance of weak learners. As a result, AdaBoost.MM does not become the original AdaBoost [4] if the number of classes is two. As [14] pointed out, the different functional form of the extended algorithm makes it difficult to interpret the extended algorithm as an extension of the statistical view of AdaBoost.

This paper shows that we can achieve boostability by invoking the same weak learnability criterion of SAMME *iteratively*. We develop the iterative weak learnability condition and demonstrate that the condition is necessary but also sufficient for boostability. Building on *iterative weak learnability* condition, we propose an efficient recursive ensemble algorithm, *AdaBoost.Iter*, retaining all desirable properties of SAMME.

We depart from the conventional view of a boosting algorithm as a process to produce a final classifier. Instead, we regard the boosting algorithm as a recursive process to eliminate under-performing classes, which allows us to extend AdaBoost to multiclass problems while retaining the functional form. We show that the probability of eliminating the correct class vanishes exponentially as the number of data increases. Consequently, the training error vanishes exponentially with the number of training samples. As *AdaBoost.Iter* retains the same functional form as AdaBoost, we can follow the same reasoning to obtain the generalization error bound that vanishes at the rate of the square root of the number of training samples.

Because the scoring function of the iterative weak learnability is linear, a linear classifier is a natural weak learner. We demonstrate that the set of linear classifiers satisfies the iterative weak learnability condition, ensuring the broad applicability of AdaBoost.Iter. We develop a *heuristic method* which quickly identifies a linear weak learner to construct the ensemble algorithm. Nevertheless, AdaBoost.Iter works well with other weak learners such as decision trees. In numerical experiments on UCI benchmark datasets, AdaBoost.Iter outperforms SAMME and achieves performance comparable to AdaBoost.MM.

The remainder of this paper is organized as follows. Section 2 illustrates the problem. Section 3 states the iterative weak learning condition. In Section 4, we show that the iterative weak learnability is necessary for boostability. In Section 5, we precisely describe AdaBoost.Iter, built on the iterative weak learnability. Section 6

shows the training error of AdaBoost.Iter vanishes at an exponential rate of the number of training data, and the generalization error vanishes at a polynomial rate of the size of training samples. Combining the two results, we conclude that AdaBoost.Iter generates a strong classifier. Section 7 reports the numerical performance of our algorithm. Section 8 concludes the paper.

2 PRELIMINARIES

Let X be an instance space, $A = 1, \ldots, |A|$ a set of classes, and $y : X \to A$ the true classification function. A probability distribution $\mathcal{D}$ is defined over the joint space of instances and labels, $X \times A$. Let $S = \{(x_1, y(x_1)), \ldots, (x_N, y(x_N))\}$ denote a finite sample set of size $N \geq 1$, where each sample is drawn independently according to the distribution $\mathcal{D}$. The empirical distribution derived from the sample set S is denoted by $\mathcal{S}$.

Let $h : X \to A$ be a weak classifier (or weak hypothesis) in the sense that $\mathbf{P}\left(h(x) \neq y(x)\right)$ may be bounded away from 0. Let $\mathcal{H}$ be the set of feasible weak classifiers.

Definition 2.1. An ensemble classifier $F^a : X \to \mathbb{R}$ is a linear combination of weak classifiers:

$$F^a(x) = \sum_k \alpha_k \mathbb{I}[h_k(x) = a] \tag{1}$$

where $h_k \subset \mathcal{H}$ and $\alpha_k \geq 0$ are the weights assigned to each weak classifier. The predicted class label for an instance x is given by

$$\arg\max_{a \in A} F^a(x).$$

Let Γ be the collection of all ensemble classifiers.

We are searching for a strong classifier F^a that is an ensemble of weak classifiers with a small probability of forecasting error

$$\mathbf{P}\left(y(x) \neq \arg\max_{a \in A} F^a(x)\right),$$

where $y(x)$ is the correct label of $x \in X$. To construct a strong classifier, the set of weak classifiers must be sufficiently rich so that we can represent the true classifier y as a linear combination of weak classifiers.

Definition 2.2. ([9], page 442) Let $y(x) \in A$ be the correct label of x. A set $\mathcal{H}$ of weak classifiers is boostable, if $\exists\{h_k\} \subset \mathcal{H}$ and $\exists\alpha_k \geq 0$ for $k \geq 1$ such that

$$F^{y(x)}(x) > F^a(x) \qquad \forall y(x) \neq a \tag{2}$$

where $F^a(x)$ is defined according to (1).

We aim to find an algorithm constructing a strong classifier by combining weak classifiers in $\mathcal{H}$. Define $s_t \in S$ as a realized sample in period t, and $O_t = (s_1, \ldots, s_{t-1})$ as a history of realized samples at the beginning of period $t \geq 1$. Define $O = \bigcup_{t \geq 1} O_t$.

Definition 2.3. τ is an ensemble algorithm if

$$\tau : O \to \Gamma$$

so that $\forall t \geq 1$, $\forall O_t$, the classifier can be represented as a linear combination of the forecasting performance of weak learners

$$\tau(O_t)(x) = \arg\max_a \sum_{k=1}^{t} \alpha_k \mathbb{I}\left[h_k(x) = a\right] \qquad \forall x. \tag{3}$$

Definition 2.4. Fix sample S. Let $T \geq 1$ be the training period. The final classifier is the output $\tau(O_T)$ at the end of training τ over S.

Definition 2.5. An ensemble algorithm τ generates a *strong forecast* if the forecasting error vanishes as the training gets longer and the sample size becomes bigger: $\forall\epsilon > 0$, $\exists\underline{m}$ such that $\forall m \geq \underline{m}$, $\exists\underline{t}$ such that if $|S| \geq \underline{m}, t \geq \underline{t}$, then $\mathbf{P}_{\mathcal{D}}(\tau(O_t)(x) \neq y(x)) \leq \epsilon$ or simply,

$$\lim_{|S| \to \infty} \lim_{t \to \infty} \mathbf{P}_{\mathcal{D}}(\tau(O_t)(x) \neq y(x)) = 0. \tag{4}$$

3 ITERATIVE WEAK LEARNABILITY

Let us illustrate SAMME [14], denoted as τ_A:

(1) Initialize the sample weights $d_1(x) = 1/|S|$ as the uniform distribution over S
(2) For $k = 1$ to K:
  (a) Fit a weak classifier $h_k \in \mathcal{H}$ to training data using weights w_k.
  (b) Compute error

$$\epsilon_k = \sum_x d_k(x) \mathbb{I}\left(h_k(x) \neq y(x)\right).$$

  (c) Calculate the weight[1]

$$\alpha_k = \frac{(|A| - 1)^2}{|A|} \log\left(\frac{(|A| - 1)(1 - \epsilon_k)}{\epsilon_k}\right) \tag{5}$$

  (d) Prepare for the next round

$$w_{k+1}(x) = \begin{cases} d_k(x) \exp\left(-(|A| - 1)\alpha_k\right) & \text{if } h_k(x) = y(x) \\ d_k(x) \exp\left(\alpha_k\right) & \text{if } h_k(x) \neq y(x). \end{cases}$$

$$d_{k+1}(x) = \frac{w_{k+1}(x)}{\sum_{x'} w_{k+1}(x')}.$$

(3) Output

$$\tau_A(O_K)(x) = \arg\max_{a \in A} \sum_{s=1}^{K} \alpha_s \mathbb{I}(h_s(x) = a) \qquad \forall x \in S$$

If $|A| = 2$, τ_A becomes AdaBoost [12]. The description of the algorithm is incomplete because if $\alpha_k < 0$, the algorithm is not well defined. For τ_A to be meaningful when $|A| = 2$, $h_k \in \mathcal{H}$ must perform better than a random guess. Given a distribution d over S,

$$\sum_{x \in S} \left(\mathbb{I}[h_k(x) = y(x)] - \mathbb{I}[h_k(x) \neq y(x)]\right) d(x) > 0,$$

which implies $\alpha_k > 0$.

3.1 Weak Learnability

The weak learnability of AdaBoost and SAMME implies the existence of h that guarantees a uniformly better performance over a random guess. Let $\Delta(S)$ be the set of all probability distributions over $S \subset X$. We state the weak learning condition according to [14].

[1]The following formula is from equation (11) on page 353 of [14], which becomes the coefficient of AdaBoost if $|A| = 2$. This formula differs slightly from (c) on page 351.

Definition 3.1. A set $\mathcal{H}$ of classifiers is weakly learnable if $\exists \rho > 0$ such that for every probability distribution $\{d(x)\}$ over $x \in S$, we have:

$$\min_{d\in\Delta(S)} \max_{h\in\mathcal{H}} \sum_{x\in S} \left(\mathbb{I}[h(x) = y(x)] \ -\mathbb{I}[h(x) \neq y(x)]\right) d(x) \geq \frac{2-|A|}{|A|} + \rho. \tag{6}$$

[9] showed that while sufficient for binary classification ($|A| = 2$), (6) is too weak to guarantee boostability if $|A| > 2$. Consider the following example in [9]:

Example 3.2. $S = \{x_1, x_2\}$, $A = \{1, 2, 3\}$, $y(x_1) = 1, y(x_2) = 2$. $\mathcal{H} = \{h_1, h_2\}$ where $h_1(x_1) = h_1(x_2) = 1$ and $h_2(x_1) = h_2(x_2) = 2$.

Since $|A| = 3$, weak learnability is satisfied if the performance of a weak classifier is strictly better than $-\frac{1}{3}$. Let $d = (d(x_1), d(x_2))$ be a probability distribution over S. The performance score of $\mathcal{H}$ is

$$\max\left[1 - 2d(x_1), -1 + 2d(x_1)\right] \geq 0$$

where equality holds if $d(x_1) = 0.5$. $\mathcal{H}$ satisfies weak learnability but not boostability, as any convex combination of h_1 and h_2 assigns the same label to both instances, while y assigns different labels.

Example 3.2 reveals that the requirement to outperform a random guess by small $\rho > 0$ becomes less restrictive when the number of classes increases. Hence, some distributions can satisfy the inequality (6) when there are many classes but can fail on a smaller subset of classes.

3.2 Iterative Weak Learnability

We strengthen weak learnability by requiring that (6) holds for any subset of A with more than a single element. We show the necessity of this condition in section 4. Let us consider a "decreasing" sequence of subsets of A: $A_I \subset \cdots \subset A_1 \subset A$ where $|A_{i+1}| < |A_i|$ for $\in \{1, \ldots, I-1\}$ and $|A_I| = 2$. Define

$$\mathcal{H}_i = \{h : X \to A_i\} \cap \mathcal{H}$$

as the collection of all weak classifiers with classes from A_i in $\mathcal{H}$.

Definition 3.3. $\mathcal{H}$ is iteratively weakly learnable if $\forall A_i \subset A$ with $|A_i| \geq 2$ and $y(x) \in A_i \ \forall x \in X$, $\exists \rho_i > 0$ such that for every distribution d over observations $x \in S$ and labels $y(x)$,

$$\min_{d\in\Delta(S)} \max_{h\in\mathcal{H}_i} \sum_{x\in S} \left(\mathbb{I}[h(x) = y(x)] \ -\mathbb{I}[h(x) \neq y(x)]\right) d(x) \geq \frac{2-|A_i|}{|A_i|} + \rho_i. \tag{7}$$

As $|A_i|$ decreases, the requirement for weak learnability of $\mathcal{H}_i$ becomes more stringent because the minimum performance criterion (the right-hand side of (7)) increases. We require that the subset A_i contains the true label $y(x)$. If the set A_i does not contain the true label, all classifiers in $\mathcal{H}_i$ have equal performance in (mis)classifying x.

In Example 3.2, $\mathcal{H}$ is weakly learnable but not iteratively weakly learnable. To see this, note that if $A_i = \{1, 2\} \subset A = \{1, 2, 3\}$, then $\mathcal{H} = \mathcal{H}_i$. Note $\min_{0\leq d(a)\leq 1} \max\left[1 - 2d(a), -1 + 2d(a)\right] = 0$. However, weak learnability over A_i requires minimum performance to be *strictly larger* than 0.

An example of the weak learners satisfying the iterative weak learnability is the collection of linear classifiers. Suppose that $x = (x_1, \ldots, x_L)$ has L features, and $A = \{a_1, \ldots, a_K\}$ is the set of K labels is the set of labels. If h is a linear classifier, $\exists(\alpha_1, \ldots, \alpha_L) \in \mathbb{R}^L$, $a^+, a^- \in A$ such that

$$h(x) = \begin{cases} a^+ & \text{if } \sum_{\ell=1}^L \alpha_\ell x_\ell \geq 0 \\ a^- & \text{if } \sum_{\ell=1}^L \alpha_\ell x_\ell < 0. \end{cases} \tag{8}$$

Let $\mathcal{H}^0$ be the set of all linear classifiers.

Proposition 3.4. *$\mathcal{H}^0$ satisfies the iterative weak learnability.*

Proof. See [2]. □

The proof of Proposition 3.4 can apply to a collection of decision trees to identify a sufficient condition that the collection of decision trees is iteratively weakly learnable.

Definition 3.5. ([3]) A hypothesis class $\mathcal{H}$ is *symmetric* if, whenever $f \in \mathcal{H}$ and $\phi : A \to A$ is a permutation, then $\phi \circ f \in \mathcal{H}$.

Lemma 3.6. *Consider an environment where the image of $y(x)$ has $|A'|$ elements where $A' \subset A$. Any symmetric hypothesis class performs at least as well as $1/|A'|$ random guessing.*

Proof. See Lemma A.1 in [2]. □

We define a sufficient condition for the iterative weak learnability.

Definition 3.7. Given $A_i \subset A$, consider a collection $\mathcal{H}_i$ of mappings $h_i : X \to A_i$. We say that a hypothesis class $\mathcal{H}_i$ is *symmetric on A_i* if, whenever $f \in \mathcal{H}_i$ and $\phi : A_i \to A_i$, there exists some $g \in \mathcal{H}_i$ such that $(\phi \circ f)(x) = g(x)$ for all $x \in X$.

Given a set of classifiers $\mathcal{H}_i$, where each $h_i \in \mathcal{H}_i$ is a mapping from X into A_i, and given $\tilde{X} \subset X$, define the set of classifiers $\mathcal{H}_i|_{\tilde{X}}$ to be the set of mappings $\tilde{h}_i : \tilde{X} \to A_i$ such that, for some $h_i \in \mathcal{H}_i$, $\tilde{h}_i(x) = h_i(x)$ for all $x \in \tilde{X}$ (i.e., $\mathcal{H}_i|_{\tilde{X}}$ is obtained from $\mathcal{H}_i$ by restricting the domain of each classifier from X to $\tilde{X}$).

We say a hypothesis class $\mathcal{H}_i$ is *symmetric on A_i when restricted to $\tilde{X}$* if, whenever $f \in \mathcal{H}_i|_{\tilde{X}}$ and $\phi : A_i \to A_i$, there exists some $g \in \mathcal{H}_i|_{\tilde{X}}$ such that $(\phi \circ f)(x) = g(x)$ for all $x \in \tilde{X}$.

Definition 3.8. We say a set of symmetric classifiers satisfies the *separation property given S* if, for any $A_i \subset A$, $\tilde{x} \in S$ and $a_0 \in A_i$, the set of classifiers $\{h \in \mathcal{H}_i|_S \mid h(\tilde{x}) = a_0\}$ contains a hypothesis class that is symmetric on A_i when restricted to $S\backslash\{\tilde{x}\}$. We say the set of symmetric classifiers satisfies the *separation property* if this holds for any training set S.

Corollary 3.9. *Suppose the training set S is finite. If $\mathcal{H}$ is symmetric and further has the separation property given S in Definition 3.8, then $\mathcal{H}$ is iteratively weakly learnable.*

Proof. Note that if $\mathcal{H}$ is iteratively weakly learnable, then so is $\tilde{\mathcal{H}} \supset \mathcal{H}$, since a larger hypothesis class only increases the left-hand side of (7). Suppose that the iterative weak learnability condition is violated for some $i \in \{2, \ldots, |A|\}$. Then, we can find some D_n such that (7) fails for $\rho_i = \frac{1}{n}$. Since $\Delta(S)$ is compact, $\exists\{D_n\}$ which converges to some limit D^*. Since $\mathcal{H}$ is symmetric, we cannot have that (7) fails when $\rho_i = 0$ by Lemma 3.6. Therefore,

$$\max_{h\in\mathcal{H}_i}\sum_{x\in S}\left(\mathbb{I}[h(x)=y(x)]-\mathbb{I}[h(x)\neq y(x)]\right)d^*(x)=\frac{2-|A_i|}{|A_i|} \quad (9)$$

It suffices to find some $h\in\mathcal{H}_i$ which, for distribution d^*, the left-hand side is larger than the right-hand side of (9). Let $\tilde{S}$ be the support of d^*, and let x_{d^*} be an extreme point of $\tilde{S}$. By the separation property and symmetry, there are classifiers that can classify x_{d^*} correctly while being symmetric on $\tilde{S}\backslash\{x_{d^*}\}$. Thus, by Lemma 3.6, (7) is satisfied if we replace S by $\tilde{S}\backslash\{x_{d^*}\}$, and replace $d^*(x)$ with $d^*(x)/(1-d^*(x_{d^*}))$. In particular, the separation property implies that the lower bound of $\frac{2-|A_i|}{|A_i|}$ is achieved on $\tilde{S}\backslash\{x_{d^*}\}$, *even while correctly classifying* x_{d^*}. It follows that, for this classifier,

$$\begin{aligned}&\sum_{x\in S}\left(\mathbb{I}[h_{d^*}(x)=y(x)]-\mathbb{I}[h_{d^*}(x)\neq y(x)]\right)d^*(x)\\&\geq d^*(x_{d^*})+\frac{2-|A_i|}{|A_i|}(1-d^*(x_{d^*}))>\frac{2-|A_i|}{|A_i|}\end{aligned}$$

contradicting (9). □

4 BOOSTABILITY

We prove that iterative weak learnability is *necessary* for boostability. Recall the definitions of $F_i^a(x)$ in (1). Define scoring function

$$\Psi_i^a(x)=(|A_i|-1)F_i^a(x)-\sum_{a'\neq a}F_i^{a'}(x)$$

where $A_i\subseteq A$ is the set of classes at the beginning of i-th iteration. In each iteration, we eliminate class $a\in A_i$ where

$$\Psi_i^a(x)<0.$$

Note that if$|A_i|=2$,

$$F_i^a(x)>0 \text{ if and only if } \Psi_i^a(x)>0$$

but the equivalent relationship fails if$|A_i|>2$.

Definition 4.1. $\mathcal{H}_i$ is *boostable* in i-th iteration or simply, *boostable*, if $\exists\{\alpha_{i,t}\}$ such that

$$F_i^{y_i(x)}(x)>F_i^a(x)\qquad\forall x,a\neq y_i(x). \quad (10)$$

We introduce a weaker notion of boostability.

Definition 4.2. $\mathcal{H}_i$ is *weakly boostable*, if $\exists\{\alpha_{i,t}\}$ such that

$$\Psi_i^{y_i(x)}(x)>0\qquad\forall x. \quad (11)$$

In the i-th iteration, we eliminate a class a with $\Psi_i^a(x)<0$, which is equivalent to eliminating a class a whose score $F_i^a(x)$ is lower than the average of other classes since

$$\Psi_i^a(x)<0 \text{ if and only if } F_i^a(x)<\frac{1}{(|A_i|-1)}\sum_{a'\neq a}F_i^{a'}(x). \quad (12)$$

Once we reach$|A_i|=2$, the weak boostability becomes equivalent to the boostability.

Proposition 4.3. *For any A_i with$|A_i|\geq 2$, boostability implies weak boostability. If$|A_i|=2$, weak boostability implies boostability.*

Proof. Note that $\Psi_i^{y_i(x)}(x)>0$ if and only if

$$|A_i|\,F_i^{y_i(x)}(x)-\sum_a F_i^a(x)>0.$$

Thus, $\mathcal{H}_i$ is weakly boostable if and only if

$$F_i^{y_i(x)}>\frac{1}{|A|}\sum_{a\in A}F_i^a(x). \quad (13)$$

The weak boostability requires that the performance of the correct label $y_i(x)$ is better than the average performance of all labels in classifying x. If $\mathcal{H}_i$ is boostable, (10) holds, which implies (13). If $|A_i|=2$, the equivalence of weak boostability and boostability follows the definition. □

We extend the notion of weak boostability to the iterative setting, where the set of classes A_i becomes smaller as i becomes larger.

Definition 4.4. $\mathcal{H}_i\subset\mathcal{H}$ is iteratively weakly boostable if (11) holds for all $A'\subset A_i$ whenever $y_i(x)\in A'\ \forall x$ and the set of weak learners is defined accordingly by restricting the range from A_i to A'.

We show that iterative weak learnability is necessary for iterative weak boostability.

Proposition 4.5. *$\mathcal{H}$ is iteratively weakly learnable if $\mathcal{H}$ is iteratively weakly boostable.*

Proof. Fix $A_I\subset\cdots\subset A_1\subset A$ with $2\leq|A_{i+1}|<|A_i|$ for $i\in\{1,\dots,I-1\}$. We show that if $\mathcal{H}$ is not iteratively weakly learnable, $\mathcal{H}$ is not iteratively weakly boostable. Suppose that $\exists i$ such that $\mathcal{H}_i$ is not iteratively weakly learnable: $\exists d^*(x)$ such that $\forall h_i\in\mathcal{H}_i$,

$$\begin{aligned}&\sum_x\left[\mathbb{I}(h_i(x)=y_i(x))-\sum_{a\neq y_i(x)}\mathbb{I}(h_i(x)=a)\right]d^*(x)\\&\qquad\leq\frac{2-|A_i|}{|A_i|}.\end{aligned} \quad (14)$$

For any $A'\subset A_i$ with $y_i(x)\in A'\ \forall x$, the associate set $\mathcal{H}'$ of weak learners is a subset of $\mathcal{H}_i$. Thus, if (14) holds for A_i, the same inequality continues to hold for any $A'\subset A_i$ with $y_i(x)\in A'\ \forall x$.

We can write (14) as $\forall h_i\in\mathcal{H}_i$,

$$\begin{aligned}\sum_x\Big[&|A_i|\,\mathbb{I}(h_i(x)=y_i(x))-|A_i|\sum_{a\neq y_i(x)}\mathbb{I}(h_i(x)=a)\\&-(2-|A_i|)\Big]d^*(x)\leq 0.\end{aligned} \quad (15)$$

A simple calculation shows that $\forall h_i\in\mathcal{H}_i$,

$$\begin{aligned}&\left[|A_i|\,\mathbb{I}(h_i(x)=y_i(x))-|A_i|\sum_{a\neq y_i(x)}\mathbb{I}(h_i(x)=a)-(2-|A_i|)\right]\\&\quad=2\left[(|A_i|-1)\mathbb{I}(h_i(x)=y_i(x))-\sum_{a\neq y_i(x)}\mathbb{I}(h_i(x)=a)\right].\end{aligned}$$

Thus, we can rewrite (15) as $h_i\in\mathcal{H}_i$,

$$2\sum_x\left[(|A_i|-1)\mathbb{I}(h_i(x)=y_i(x))-\sum_{a\neq y_i(x)}\mathbb{I}(h_i(x)=a)\right]d^*(x)\leq 0.$$

For any sequence $\{\alpha_{i,t}\}$, and for any $\{h_{i,t}\} \in \mathcal{H}_i$,

$$\begin{aligned}
0 &\geq 2\sum_t \alpha_{i,t}\Bigg(\sum_x \Big[(|A_i|-1)\mathbb{I}(h_{i,t}(x)=y_i(x)) \\
&\quad - \sum_{a\neq y_i(x)} \mathbb{I}(h_{i,t}(x)=a)\Big]d^*(x)\Bigg) \\
&= 2\sum_x \Big[(|A_i|-1)\sum_t \alpha_{i,t}\mathbb{I}(h_{i,t}(x)=y_i(x)) \\
&\quad - \sum_{a\neq y_i(x)}\sum_t \alpha_{i,t}\mathbb{I}(h_{i,t}(x)=a)\Big]d^*(x) \\
&= 2\sum_x \Big[(|A_i|-1)F_i^{y_i(x)}(x) - \sum_{a\neq y_i(x)} F_i^a(x)\Big]d^*(x) \\
&= 2\sum_x \Psi_i^{y_i(x)}(x)d^*(x).
\end{aligned}$$

Therefore, $\exists x \in X$ with $d^*(x) > 0$ so that $\Psi^{y_i(x)}(x) \leq 0$, implying that $\mathcal{H}_i$ is not iteratively weakly boostable. □

5 ALGORITHM CONSTRUCTION

To show that the iterative weak learnability is sufficient for the iterative weak boostability, we construct an algorithm, called AdaBoost.Iter, that generates a strong classifier, using $\Psi_i^a(x)$ as the scoring function for the weak learners. AdaBoost.Iter iteratively eliminates classes that perform worse than the average performance of the classes until the number of classes becomes 2. If $|A_i| = 2$, AdaBoost.Iter becomes AdaBoost, generating a strong classifier.

5.1 Overview

AdaBoost.Iter iterates over two loops: within an epoch (inner loop, Algorithm 1) and across epochs (outer loop, Algorithm 2). The length of each epoch is $K \geq 1$, which is a parameter we choose to meet the performance requirement of the algorithm. Let $k = 1, 2, \ldots, K$ denote the iteration index within each epoch and $i = 1, 2, \ldots$ denote the epoch index. We use the subscript $_{i,k}$ to refer to the k-th iteration of the i-th epoch.

Epoch i starts with the set of classes $A_i = \{1, \ldots, |A_i|\}$ where $A_i \subset A$ with $A_1 = A$ which is the first $|A_i|$ positive integers. By the end of epoch i, the algorithm identifies a class $b_i(x) \in A_i$ that will be eliminated at the end of K-th round in epoch i, as we illustrated in Section 5.2. Even though $|A_{i+1}| = |A_i| - 1$, we need to construct A_{i+1} from $A_i \setminus \{b_i(x)\}$, described in Section 5.3. The algorithm has I number of epochs. The number of epochs depends on whether we delete one or more classes from A in each epoch.

Define $I = \min\{i \,|\, |A_i| = 2\}$ as the first epoch when the number of classes is 2. By the end of epoch I, the algorithm stops and produces the final classifier. The total number of iterations is thus $KI = K(|A| - 1)$ if one class is eliminated per epoch. We can also consider eliminating multiple classes in each epoch.

5.2 Within Epoch i

See Algorithm 1 for the pseudo-code for the process within an epoch. Define $A_1 = A$ and $y_1(x) = y(x)$ as the correct label in $A_1 = A$. Epoch $i \geq 1$ starts with $A_i \subset A$ and $y_i : S \to A_i$ which is the correct label in epoch i. Set $k = 1$, and set $d_{i,1}(x)$ as the uniform distribution over S.

$$d_{i,1}(x) = \frac{1}{|S|} \qquad \forall x \in S.$$

The output of each stage consists of three key components: an artificial probability distribution $d_{i,k}(x)$ over S, a weak classifier $h_{i,k}$ in step k, and a positive weight $\alpha_{i,k}$. Suppose that $d_{i,k}(x)$ is defined $\forall x \in S$. Choose weak learner $h_{i,k}$ satisfying the iterative weak learnability condition (7).

Define

$$\epsilon_{i,k} = \mathbf{P}_{d_{i,k}}\Big(h_{i,k}(x) \neq y_i(x)\Big) \tag{16}$$

as the probability that the optimal classifier $h_{i,k}$ at step k misclassifies x under $d_{i,k}$. If $\epsilon_{i,k} = 0$, then we stop the training and output $h_{i,k}$ as the final classifier, which perfectly forecasts $y_i(x)$.

Suppose that $\epsilon_{i,k} > 0$. Define

$$\alpha_{i,k} = \frac{1}{2(|A_i|-1)}\left[\log\frac{(|A_i|-1)(1-\epsilon_{i,k})}{\epsilon_{i,k}}\right]. \tag{17}$$

Define for each x, and each pair $(x, y_i(x))$,

$$d_{i,k+1}(x) = \frac{d_{i,k}(x)\Lambda_{i,k}(x)}{Z_{i,k}}$$

where

$$\begin{aligned}
&\Lambda_{i,k}(x) \\
&= \exp\Big(-\alpha_{i,k}\left[(|A_i|-1)\mathbb{I}(h_{i,k}(x)=y_i(x)) - \mathbb{I}(h_{i,k}(x)\neq y_i(x))\right]\Big)
\end{aligned}$$

and

$$Z_{i,k} = \sum_{x\in S} d_{i,k}(x)\Lambda_{i,k}(x).$$

Given $d_{i,k+1}$, we can recursively define $h_{i,k+1}$ and $\epsilon_{i,k+1}$.

Define $\forall a \in A_i = \{1, \ldots, |A_i|\}$,

$$F_{i,k}^a(x) = \sum_{s=1}^k \alpha_{i,s}\mathbb{I}(h_{i,s}(x)=a)$$

and

$$\Psi_{i,k}^a(x) = (|A_i|-1)F_{i,k}^a(x) - \sum_{a'\neq a} F_{i,k}^{a'}(x). \tag{18}$$

A simple calculation shows that $\forall x$,

$$a \in \arg\max_{a'\in A_i} F_{i,k}^{a'}(x) \tag{19}$$

if and only if

$$a \in \arg\max_{a'\in A_i} \Psi_{i,k}^{a'}(x). \tag{20}$$

Since $\Psi_{i,k}^a(x)$ is a linear combination of $\mathbb{I}(h_{i,s}(x) = a)$, our algorithm is an ensemble algorithm.

Since $\sum_{a\in A_i}\Psi_{i,K}^a(x) = 0$, $\exists a \in A_i$, $\Psi_{i,K}^a(x) \leq 0$. Thus, if $b_i(x) \in A_i$ satisfies

$$\Psi_{i,K}^{b_i(x)}(x) \leq \Psi_{i,K}^a(x) \qquad \forall a \in A_i,$$

$\Psi_{i,K}^{b_i(x)}(x) \leq 0$ $\forall x$. The algorithm removes $b_i(x)$ from A_i.[2]

[2] If multiple minimizers exist, we choose one according to a fixed rule.

5.3 Between Epochs

Note that $b_i(x) \neq b_i(x')$ if $x \neq x'$ in general and that a boosting algorithm requires that each instance $x \in X$ is endowed with the identical set of classes. Since the set of remaining classes after elimination, $A_i \setminus \{b_i(x)\}$, can differ across different instances, we need to construct A_{i+1} by "homogenizing" the set of classes across different instances before epoch $i+1$ starts.

See Algorithm 2 for the pseudo-code for the process between the epochs. Algorithm 2 eliminates underperforming classes and "homogenizes" the set of remaining classes, recursively generating $\{A_i\}$, $\{\mathsf{Elim}_i(x)\}_x$, $\{\mathsf{Sur}_i(x)\}_x$, and $\lambda_i^x(a)$ where $i \in \{1,\ldots,|A|-1\}$ defined as follows.

- $\mathsf{Elim}_i(x)$ is the list of labels eliminated before epoch i. Naturally, $\mathsf{Elim}_1(x) = \emptyset$. $\mathsf{Elim}_i(x)$ records which label is eliminated and when the label is eliminated.
- $\mathsf{Sur}_i(x)$ is the list of labels in A that have survived the elimination process before epoch i. Define $\mathsf{Sur}_1(x) = A\ \forall x$. Clearly, $\mathsf{Elim}_i(x)$ and $\mathsf{Sur}_i(x)$ partitions A in every epoch i.
- The surviving labels are placed in the front, while the eliminated labels are stacked at the back of the list. If $\lambda_i^x(a) \leq \left|\mathsf{Sur}_i(x)\right|$, then $a \in \mathsf{Sur}_i(x)$. Otherwise, a is eliminated before epoch i.

Given $b_i(x)$, we update Elim_i, Sur_i and A_i as follows.

$$\begin{aligned} \mathsf{Elim}_{i+1}(x) &= \mathtt{Concat}(b_i(x), \mathsf{Elim}_i(x)) \\ \mathsf{Sur}_{i+1}(x) &= \mathtt{Sorted}(\mathsf{Sur}_i(x) \setminus \{b_i(x)\}) \\ A_{i+1} &= (1,\ldots,\left|\mathsf{Sur}_{i+1}(x)\right|) \qquad \forall x. \end{aligned}$$

Epoch $i+1$ starts with each sample endowed with $A_{i+1}\ \forall x$. We eliminate a single class in each epoch. By the construction, $|A_{i+1}| = |A_i| - 1$ and $\left|\mathsf{Elim}_{i+1}(x)\right| = \left|\mathsf{Elim}_i(x)\right| + 1$ for each x. [3]

5.4 Final Classifier

Let us write the constructed algorithm τ^K_{Iter} because the algorithm is parameterized by the minimum length of each epoch K. We refer to τ^K_{Iter} as AdaBoost.Iter when we compare it to other algorithms. The final classifier is

$$\tau^K_{\mathsf{Iter}}(O_{KI})(x) = \arg\max_{a\in A} \sum_{i=1}^{|A|-1} \mathbb{I}\left(\Psi^{\lambda_i^x(a)}_{i,K}(x) > 0\right).$$

The algorithm searches for $a \in A$ to maximize the right-hand side, which survives the elimination process for all epochs. If $|A| = 2$, $I = 1$ and τ^K_{Iter} is the AdaBoost algorithm. We shall select K to satisfy the final classifier's accuracy and confidence requirement.

[3] We choose to eliminate exactly one class in each iteration to simplify the illustration of AdaBoost.Iter. We can improve the algorithm's performance by eliminating more than one class in each iteration. Although the procedure is defined for the training sample, the algorithm is well defined for any $x \in X$, including the testing samples. Given sample $\mathcal{S}$, $\{\alpha_{i,t}\}$ and $\{h_{i,t}\}$ are determined. For each $a \in A_i$, $\Psi^a_{i,K}(x)$ is well defined for $x \in X$, since the domain of $h_{i,t}$ is X. Once $\Psi^a_{i,K}(x)$ is determined, the housekeeping tasks depend solely on $\Psi^a_{i,K}(x)$. We can identify $\{b_i(x)\}_i$ for any testing sample $x \in X$ in epoch i. We can recursively define $\lambda_i^x(a)$ (along with $\mathsf{Sur}_i(x)$ and $\mathsf{Elim}_i(x)$) for given $b_i(x)$. Therefore, the algorithm produces an unambiguous "out of sample" forecast for an arbitrary $x \in X$, even if $x \in X \setminus S$.

Algorithm 1 Boost Phase

Input: Training set $S = \{(x_1,y_1),\ldots,(x_{|S|},y_{|S|})\}$, number of iterations K, class labels A_i
Output: Ensemble of weak learners $\{h_{i,1},\ldots,h_{i,K}\}$, their weights $\{\alpha_{i,1},\ldots,\alpha_{i,K}\}$
Initialize the sample weights $d_{i,1}(x) \leftarrow 1/|S|$
for $k = 1,\ldots,K$ **do**
Fit a weak learner $h_{i,k}$ to the training set using weights d_i
Compute error

$$\epsilon_{i,k} \leftarrow \sum_s d_{i,k}(x)\mathbb{I}(h_{i,k}(x) \neq y_i(x))$$

Compute the weight of $h_{i,k}$

$$\alpha_{i,k} \leftarrow \frac{1}{2(|A_i|-1)} \log \frac{(|A_i|-1)(1-\epsilon_{i,k})}{\epsilon_{i,k}}$$

Update the weights of the training set

$$\begin{aligned} d_{i,k+1}(x) \propto & d_{i,k}(x) \exp\Big\{\alpha_{i,k}\big[\mathbb{I}(h_{i,k}(x) \neq y_i(x)) \\ & -(|A_i|-1)\mathbb{I}(h_{i,k}(p) = y_i(p))\big]\Big\}. \end{aligned}$$

Compute

$$F^a_{i,k}(x) \leftarrow \sum_{j=1}^{k} \alpha_{i,j}\mathbb{I}(h_{i,j}(x) = a)$$

$$\Psi^a_{i,k}(x) \leftarrow (|A_i|-1)F^a_{i,k}(x) - \sum_{a'\neq a} F^{a'}_{i,k}(x)$$

Compute

$$u_{i,k}(p) = |\{a|\Psi^a_{i,k}(p) < 0\}|$$

end for

Algorithm 2 Elimination Phase

Input: Set of all classes A, training set S
Output: Final predictor $\tau^K_{\mathsf{Iter}}(x)$
$A_1 \leftarrow A$, $\mathsf{Sur}_1 \leftarrow A$, and $\mathsf{Elim}_1 \leftarrow \emptyset$
while $i \leq |A| - 1$ **do**
Run Algorithm 1, and generate Ψ^a_{i,K_i}
$b_i(x) \leftarrow \max\{a \in \mathsf{Sur}_i(x)|\Psi^a_{i,K}(x) < 0\}$
$\mathsf{Sur}_{i+1}(x) \leftarrow \mathsf{Sur}_i(x) \setminus b_i(x)$
$\mathsf{Elim}_{i+1}(x) \leftarrow [b_i(x), \mathsf{Elim}_i(x)]$
$\lambda^a_{i+1}(x) \leftarrow l : \mathsf{Sur}_{i+1}(x)[l] = a$
$A_{i+1} \leftarrow [1, ..., |A_i| - 1]$
$i \leftarrow i + 1$
end while
Final predictor $\tau^K_{\mathsf{Iter}}(x) = \arg\max_a \sum_i \mathbb{I}\left(\Psi^{\lambda_i^a(x)}_{i,K} > 0\right)$

6 PERFORMANCE

Definition 6.1. Algorithm τ is efficient if $\exists\rho > 0$, $\exists K > 0$ such that $\forall k \geq K$,

$$\mathbf{P}_{\mathcal{S}}\left(\tau(O_k)(x) \neq y(x)\right) \leq e^{-\rho k} \qquad \forall \mathcal{S}. \tag{21}$$

AdaBoost.Iter is an efficient algorithm where the training error vanishes exponentially.

Theorem 6.2. *Assume that $|A| < \infty$ and $|S| < \infty$, and that $\mathcal{H}$ is iteratively weakly learnable. $\exists \underline{\rho} > 0$ and $\overline{K}$ such that $\forall K \geq \overline{K}$,*

$$\mathbf{P}\left(\tau^K_{\mathsf{Iter}}(O_{KI})(x) \neq y(x)\right) \leq |A|\, e^{-\underline{\rho} K}.$$

Proof. See Appendix A. □

We obtain bounds on the generalization error based on the classification margin. AdaBoost [12] stops at the end of a single epoch for binary classification to produce the final classifier. [12] obtain a pair of results on the generalization error:[4] one for the case of $|\mathcal{H}| < \infty$ and one where $|\mathcal{H}|$ may be infinite but having finite VC-dimension. We state the generalization error in epoch i in the same format as [12]. To do this for the case where $|\mathcal{H}| = \infty$, we use the Graph dimension (as in [10]) in place of VC-dimension.[5]

Theorem 6.3. *Fix epoch i endowed with $A_i \subset A$ with $y_i(x) \in A_i$. Assume d^* is a distribution over $X \times A_i$ and let S be a sample of m examples chosen independently at random according to $\mathcal{D}$.*

Assume that the set $\mathcal{H}_i$ of weak learners is finite, and let $\delta > 0$. Then with probability at least $1 - \delta$ over the random choice of the training set S, $\Psi^{y_i(x)}_{i,K}(x)$ satisfies the following bound:

$$\mathbf{P}_{\mathcal{D}}[\Psi^{y_i(x)}_{i,K}(x) \leq 0] \leq \mathbf{P}_{\mathcal{S}}[\Psi^{y_i(x)}_{i,K}(x) \leq \theta] + \mathrm{O}\left(\sqrt{|A_i|^2 \frac{\log|\mathcal{H}_i|}{m\theta^2} \log\left(\frac{m\theta^2}{\log|\mathcal{H}_i|}\right) + \frac{\log(1/\delta)}{m}}\right),$$

for all $\theta > \sqrt{(\ln|\mathcal{H}_i|)/(4m)}$. Suppose that, instead of being finite, $\mathcal{H}_i$ has Graph dimension d_i, with $\delta > 0$ and $m \geq d_i \geq 1$. Then, with probability at least $1 - \delta$ over the random choice of the training set S, every such $\Psi^{y_i(x)}_{i,K}(x)$ satisfies the following bound

$$\mathbf{P}_{\mathcal{D}}[\Psi^{y_i(x)}_{i,K}(x) \leq 0] \leq \mathbf{P}_{\mathcal{S}}[\Psi^{y_i(x)}_{i,K}(x) \leq \theta] + \mathrm{O}\left(\sqrt{\frac{|A_i|^2 d_i \log(m/d_i)\log(m\theta^2/d_i)}{m\theta^2} + \frac{\log(1/\delta)}{m}}\right),$$

for all $\theta > \sqrt{8 d_i \ln(em/d_i)/m}$.

Proof. See Appendix B. □

Transforming the result from individual epochs into an overall bound on the probability of misclassification is a straightforward application of the union bound.

Corollary 6.4. *Let $\mathcal{D}$ be a distribution over $X \times A$ and let S be a sample of m examples chosen independently at random according to $\mathcal{D}$. Assume that the base classifier space $\mathcal{H}$ is finite, and let $\delta > 0$. Then with probability at least $1 - \delta$ over the random choice of the training set S,*

$$\mathbf{P}_{\mathcal{D}}[\exists i,\ \Psi^{y_i(x)}_{i,K_i}(x) \leq 0] \leq |A| \max_{i\in\{1,\ldots,I\}} \mathbf{P}_{\mathcal{S}}[\Psi^{y_i(x)}_{i,K_i}(x) \leq \theta] + \mathrm{O}\left(|A| \sqrt{|A|^2 \frac{\log|\mathcal{H}|}{m\theta^2} \log\left(\frac{m\theta^2}{\log|\mathcal{H}|}\right) + \frac{\log(1/\delta)}{m}}\right),$$

for all $\theta > \sqrt{(\ln|\mathcal{H}|)/(4m)}$.

Suppose that, instead of being finite, $\mathcal{H}$ has Graph dimension d, with $\delta > 0$ and $m \geq d \geq 1$. Then, with probability at least $1 - \delta$ over the random choice of the training set S,

$$\mathbf{P}_{\mathcal{D}}[\exists i, \Psi^{y_i(x)}_{i,K}(x) \leq 0] \leq |A| \max_{i\in\{1,\ldots,I\}} \mathbf{P}_{\mathcal{S}}[\Psi^{y_i(x)}_{i,K}(x) \leq \theta] + \mathrm{O}\left(|A| \sqrt{\frac{|A|^2 d \log(m/d)\log(m\theta^2/d)}{m\theta^2} + \frac{\log(1/\delta)}{m}}\right),$$

for all $\theta > \sqrt{8d \ln(em/d)/m}$.

7 EXPERIMENTAL RESULTS

To illustrate our algorithm's performance and practical properties, we explore the numerical performance of AdaBoost.Iter in the synthetic example in [14] and the benchmark datasets from the University of California, Irvine (UCI) repository [8]. [6]

7.1 Synthetic Data

The first exercise replicates the synthetic data exercise depicted in Figure 1 on page 351 in [14]. The feature vector, x, is drawn from a ten-dimensional standard normal distribution for each instance. The true class, y, of each instance is assigned according to:

$$y_i = \begin{cases} 1 & \text{if } 0 \leq \sum_{j=1}^{10} x_j^2 < \chi^2_{10,\frac{1}{3}}, \\ 2 & \text{if } \chi^2_{10,\frac{1}{3}} \leq \sum_{j=1}^{10} x_j^2 < \chi^2_{10,\frac{2}{3}}, \\ 3 & \text{otherwise.} \end{cases} \tag{22}$$

where $\chi^2_{10,p}$ is the p quantile of 10 dimensional χ^2 distirubtuion. Instances are designed to be cast into a set of nested concentric spheres. The training set consists of 3,000 instances, and the test set has 10,000 instances.

On the synthetic dataset, we compare the performance of the AdaBoost.Iter and AdaBoost.MM [9], as both algorithms are built on the notion of weak learnability that is equivalent to boostability. We implemented the weak learner algorithm for AdaBoost.MM following the description in [9] while we use linear classifiers as weak learners for AdaBoost.Iter. The choice of linear classifiers in AdaBoost.Iter stems from the linear nature of our weak learnability criterion. We identify a weak learner according to a heuristic weak learner algorithm described in Algorithm 3. The heuristic method finds a linear classifier satisfying (6) begins by lopping through all possible 2-class combinations from the set of classes. The pair of classes and the hyperplane with the highest classification accuracy are selected.

Because the iterative weak learnability is weaker than EOR of AdaBoost.MM, the selected linear classifiers by the heuristic weak learner algorithm in AdaBoost.Iter might reduce the forecasting error less than those in AdaBoost.MM. However, it is much faster to identify a weak learner in AdaBoost.Iter than AdaBoost.MM, thanks to the linear scoring function of the iterative weak learnability. Even though the final classifier of AdaBoost.Iter may entail more

[4] originally presented in [13]

[5] See also [1] and [7] for statements of this definition.

[6] In all of these experiments. We conducted experiments on a MacBook Pro equipped with a 2.4 GHz 8-core Intel Core i9 processor, 16 GB of DDR4 RAM running at 2400 MHz, and a 512 GB SSD for storage.

Algorithm 3 Heuristc weak learner algorithm

Generate a list, $L = \{l_j\}$, of all 2-class combination out of A
for l_j in L **do**
Resample, $\{\tilde{y}_p, \tilde{X}_p\}$, from $\{y_p, X_p\}$ with replacement by weight $\{d_p\}$
Run OLS regression on subsample of $\{\tilde{y}_p, \tilde{X}_p\}$ where $\tilde{y}_p \in l_j$
Compute OLS coefficient $\beta_j = \left(\tilde{X}_p'\tilde{X}_p\right)^{-1}\tilde{X}_p'\tilde{y}_p$, and rate of correct prediction

$$\text{fit}_j \leftarrow \sum_p d_p \left(y_p == l_{j,1}\mathbb{I}\left(X\beta_j \geq 0\right) + l_{j,2}\mathbb{I}\left(X\beta_j < 0\right)\right)$$

end for
$J \leftarrow \arg\max_j \text{fit}_j$
$\text{Fit} \leftarrow \text{fit}_J$
$\beta \leftarrow \beta_J$
if $\text{Fit} \leq \frac{2-|A|}{|A|}$ **then**
$J \leftarrow \arg\max_j \sum_p d_p \mathbb{I}(y_p \in \{l_j\})$
$\beta \leftarrow -\beta_J$
end if

weak classifiers than that of AdaBoost.MM, AdaBoost.Iter saves significant time in identifying a suitable weak learner to produce a strong classifier much faster, sometimes as much as seven times faster than AdaBoost.MM.

We compare the AdaBoost.Iter and AdaBoost.MM in terms of (1) the error rate of algorithms given the computational runtime, (2) the computation runtime, and the number of weak learners ensembled to reach some threshold of training error rate.

Table 1: Computation runtime and the number of weak learners for reaching different training error thresholds on a synthetic dataset. Linear classifiers are used as weak classifiers for AdaBoost.Iter and depth-three trees are used for AdaBoost.MM.

Error	AdaBoost.Iter	AdaBoost.MM
Threshold(%)	Runtime (s)	
10	47.7	733.5
5	98.5	1586.8
1	481.9	4589.3
	# of Weak Learners	
10	3500	164
5	5500	243
1	13700	421

Table 1 records the number of weak learners, the runtime, and the testing errors when the training error reaches a predetermined level. Since both algorithms are boostable, the training errors converge to 0. Table 1 shows that AdaBoost.It takes significantly less time than AdaBoost.MM to identify a linear classifier to construct a final classifier because AdaBoost.Iter uses a simple linear rule to calculate the score. As a result, AdaBoost.Iter needs many more weak learners but takes less time to produce the final classifier. In this synthetic example, AdaBoost.Iter can be trained to achieve the desired forecast accuracy between 6.5% and 10.5% of AdaBoost.MM's runtime.

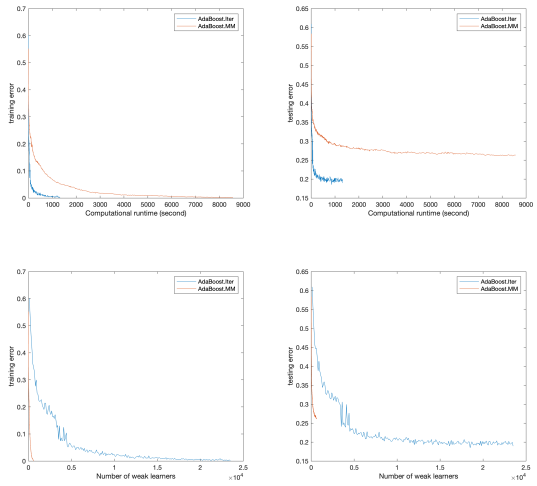

Figure 1: Comparing AdaBoost.Iter with AdaBoost.MM. Specifications of the data generation are the same as those used to generate Figure 1 in [14]. The linear classifiers are used as weak classifiers for AdaBoost.Iter and the depth of three trees are used as weak classifiers for AdaBoost.MM.

The left panels in Figure 1 report how the training errors vanish with respect to the runtime (upper left panel) and the number of weak learners (lower left panel). Even though AdaBoost.MM requires fewer weak learners to construct a final classifier, it takes a considerably longer time to identify an optimal weak learner (average 8.8 vs. 0.06 seconds per weak learner), contributing to a significantly longer runtime of AdaBoost.MM than AdaBoost.Iter (7750 vs. 1554 seconds). Note that the testing errors of both algorithms are stabilized. The right panels in Figure 1 show how the testing errors decrease with respect to the run time (upper right panel) and the number of weak learners (lower right panel).

7.2 UCI Data

This section summarizes the performance of AdaBoost.Iter, AdaBoost.MM, SAMME, and Gradient Boosting [6] on UCI datasets. We used 9 datasets summarized in Table 2. When the training and testing set split is not available, 20% of the examples were randomly selected as the testing set. The algorithms are trained on the training set, and their performance is evaluated on the test set. We use the same weak learner, the decision tree stump, to compare the four algorithms fairly. The implementation of SAMME and Gradient Boosting is from the scikit-learn package [11].

Table 3 lists the testing errors across the chosen UCI datasets and standard deviations in brackets derived via Bootstrap. The results in Table 3 show that AdaBoost.Iter and AdaBoost.MM consistently outperforms SAMME on all datasets. AdaBoost.Iter performs best

Table 2: Summary of benchmark datasets

Dataset	Train size	Test size	Features	Classes
Landsat Satellite	4435	2000	36	6
Image Segmentation	210	2100	19	7
Letter	16000	4000	16	26
Pen Digit	7494	3498	16	10
Poker	25010	1000	10	10
Optical Digit	4496	1124	64	10
Vehicle	676	170	18	4
Ecoli	268	68	7	8
Vertebral	248	62	6	3

on 3 out of 9 datasets (Landsat Satellite, Letter, and Vehicle), while AdaBoost.MM performs best on 5 datasets (Image Segmentation, Pen Digit, Poker, Optical Digit, and Ecoli). However, except for Image Segmentation and Poker, the differences in performance between the two algorithms are relatively small, with overlapping standard deviations.

8 CONCLUSION

The weak learnability of SAMME is too weak to learn the true classifier except for binary classification, as pointed out by [9]. Our innovation is to invoke weak learnability iteratively while eliminating "poorly" performing classes until $|A_i|$ becomes binary. We achieve boostability by combining the weak learnability with the iterative elimination process. As τ^K_{Iter} is built on the framework of SAMME, the algorithm inherits the same statistical foundation as SAMME and is a direct extension to the AdaBoost algorithm for multiclass classification problems.

ACKNOWLEDGMENTS

We thank Shaowei Ke, Tim Roughgarden, and Rob Schapire for their helpful comments. The first author gratefully acknowledges the financial support from the National Science Foundation (SES-1952882) and is grateful for the hospitality and support from the University of Southern California.

Table 3: Testing errors and standard deviations for different algorithms on benchmark datasets. The decision tree stumps are used as weak classifiers.

Dataset	AdaBoost.Iter	AdaBoost.MM	SAMME	Gradient Boosting
Landsat Satellite	**0.1040** (0.0070)	0.1135 (0.0077)	0.2185 (0.0088)	0.1255 (0.0087)
Image Segmentation	0.0824 (0.0061)	**0.0529** (0.0048)	0.1038 (0.0062)	0.0662 (0.0048)
Letter	**0.1003** (0.0050)	0.1230 (0.0055)	0.4928 (0.0072)	0.1250 (0.0062)
Pen Digit	0.0392 (0.0033)	**0.0369** (0.0030)	0.2642 (0.0073)	0.0486 (0.0034)
Poker	0.4560 (0.0160)	**0.4020** (0.0151)	0.5390 (0.0157)	0.5060 (0.0150)
Optical Digit	0.0258 (0.0048)	**0.0222** (0.0040)	0.0934 (0.0092)	0.0338 (0.0056)
Vehicle	**0.1941** (0.0271)	0.2118 (0.0269)	0.3588 (0.0348)	0.2412 (0.0330)
Ecoli	0.1471 (0.0404)	**0.1324** (0.0393)	0.2353 (0.0499)	0.1765 (0.0389)
Vertebral	**0.1452** (0.0382)	**0.1452** (0.0418)	0.2097 (0.0530)	0.1774 (0.0487)

REFERENCES

[1] Shai Ben-David, Nicolo Cesa-Bianchi, David Haussler, and Philip M. Long. 1995. Characterizations of Learnability for Classes of $\{0, ..., n\}$-valued Functions. *J. Comput. System Sci.* 50, 1 (1995), 74–86.

[2] In-Koo Cho and Jonathan Libgober. 2021. Machine learning for strategic inference. *arXiv preprint arXiv:2101.09613* (2021).

[3] Amit Daniely, Sivan Sabato, Shai Ben-David, and Shai Shalev-Shwartz. 2015. Multiclass Learnability and the ERM Principle. *Journal of Machine Learning Research* 16 (December 2015), 2377–2404.

[4] Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences* 55, 1 (1997), 119–139.

[5] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2000. Additive Logistic Regression: A Statistical View of Boosting. *The Annals of Statistics* 28, 2 (2000), 337–407.

[6] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[7] Yann Guermeur. 2007. VC Theory of Large Margin Multi-Category Classifiers. *Journal of Machine Learning Research* 8 (November 2007), 2551–2594.

[8] J Mertz and PM Murphy. 2005. University of California at Irvine (UCI) repository of machine learning databases. *Available ftp://ftp. ics. uci. edu/pub/machine-learning-databases* (2005).

[9] Indraneel Mukherjee and Robert E. Schapire. 2013. A Theory of Multiclass Boosting. *Journal of Machine Learning Research* 14 (2013), 437–497.

[10] Balas K. Natarajan. 1989. On learning sets and functions. *Machine Learning* 4, 1 (1989), 67–97.

[11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

[12] Robert E. Schapire and Yoav Freund. 2012. *Boosting: Foundations and Algorithms.* MIT Press.

[13] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. 1998. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics* 26, 5 (October 1998), 1651–1686.

[14] Ji Zhu, Hui Zou, Saharon Rosset, and Trevor Hastie. 2009. Multi-class AdaBoost. *Statistics and Its Interface* 2 (2009), 349–360.

A PROOF OF THEOREM 6.2

Uniform Bound for I.

We claim $I \leq \min(|A|, |S|)$. Let us focus on the first epoch: $i = 1$. Recall that $\forall d_{1,k}$, $h_{1,k}$ is maximizing

$$\sum_{x \in S} \left(\mathbb{I}[h_1(x) = y_1(x)] - \mathbb{I}[h_1(x) \neq y_1(x)] \right) d_k(x).$$

Let $y_1(S)$ be the collection of all true labels. If $\exists x \in S$ such that $h_1(x) = a \in A \setminus y(S)$, then h_1 cannot be an optimal weak classifier. To see this, suppose that $a \in A \setminus y(S)$ such that $h_1(x) = a$. The decision maker can increase the performance by replacing a by $a^+ \in y_i(S)$. Thus, $\forall k$, $h_{1,k}(S) \subset y_1(S)$.

Recall that $\forall a \in A_i$,

$$F^a_{i,k}(x) = \sum_{s=1}^{k} \alpha_{i,s} \mathbb{I}(h_{i,s}(x) = a)$$

and

$$\Psi^a_{i,k}(x) = \sum_{s=1}^{k} \alpha_{i,s} \left(|A_i| \, \mathbb{I}(h_{i,s}(x) = a) - 1 \right).$$

Thus,

$$\begin{aligned} \sum_a \Psi^a_{i,k}(x) &= \sum_a \left(\sum_{s=1}^{k} \alpha_{i,s} \left[|A_i| \, \mathbb{I}(h_{i,s}(x) = a) - 1 \right] \right) \\ &= \sum_{s=1}^{k} \alpha_{i,s} \left[|A_i| - |A_i| \right] = 0. \end{aligned}$$

We have $\forall k$, $\forall a \notin y_1(S) \subset A$,

$$\Psi^a_{1,k}(x) = \sum_{s=1}^{k} \alpha_{1,s} \left(|A| \, \mathbb{I}(h_{1,s}(x) = a) - 1 \right) = -\sum_{s=1}^{k} \alpha_{1,s} < 0.$$

Every $a \in A \setminus y(S)$ is eliminated by the end of the first epoch. Since $|y(S)| \leq |S|$, the number of remaining labels after the first epoch is no more than $\min(|A|, |S|)$. **Within Epoch.** We show that if the minimum length K of an epoch is sufficiently large, then $\exists b_i(x) \in A_i$ such that $\Psi^{b_i(x)}_{i,K}(x) < 0 \ \forall x$. We calculate the lower bound of the probability: $\exists \rho_i > 0$ such that the probability is bounded from below by $1 - e^{-\rho_i K}$.

Iterative weak learnability implies that $\forall i$, $\exists \gamma_i$, $\forall d_{i,k}$, $\exists h_{i,k} \in \mathcal{H}_i$ so that

$$\sum_{x \in S} \left(\mathbb{I}[h_k(x) = y_i(x)] - \mathbb{I}[h_k(x) \neq y_i(x)] \right) d(x) \geq \frac{2 - |A_i|}{|A_i|} + \gamma_i.$$

Lemma A.1.

$$\mathbf{P}\left(\Psi^{y_i(x)}_{k,i}(x) \leq 0 \right) \leq e^{-\rho_i k}.$$

Proof. Note that

$$\begin{aligned} \mathbf{P}\left(\Psi^{y_i(x)}_{i,k} \leq 0 \right) &= \mathbf{E}_{d_{i,1}} \mathbb{I}\left(\Psi^{y_i(x)}_{i,k}(x) \leq 0 \right) \\ &\leq \mathbf{E}_{d_{i,1}} \exp\left(-\Psi^{y_i(x)}_{i,k}(x) \right). \end{aligned} \tag{23}$$

From the updating formula of $d_{i,k}$,

$$\begin{aligned} d_{i,k+1}(x) &= \frac{d_{i,1}(x) \exp\left[-(|A_i| - 1) F^{y_i(x)}_{i,k}(x) + \sum_{a \neq y_i(x)} F^a_{i,k}(x) \right]}{\prod_{s=1}^{k} Z_{i,s}} \\ &= \frac{d_{i,1}(x) \exp\left[-\Psi^{y_i(x)}_{i,k}(x) \right]}{\prod_{s=1}^{k} Z_{i,s}}. \end{aligned} \tag{24}$$

By summing over x, we have

$$\sum_x d_{i,1}(x) \exp\left[-\Psi^{y_i(x)}_{i,k}(x) \right] = \prod_{s=1}^{k} Z_{i,s}.$$

Note $\forall k \geq 1$, $\forall s \{1, \ldots, k\}$,

$$\begin{aligned} & Z_{i,s} \\ = & \sum_{h_{i,s}(x) = y_i(x)} d_{i,s}(x) e^{-(|A_i|-1)\alpha_{i,s}} + \sum_{h_{i,s}(x) \neq y_i(x)} d_{i,s}(x) e^{\alpha_{i,s}} \\ = & \ (|A_i| - 1)^{-\frac{1}{2}} \epsilon_{i,s}^{\frac{1}{2}} (1 - \epsilon_{i,s})^{\frac{1}{2}} \\ & + (|A_i| - 1)^{\frac{1}{2(|A_i|-1)}} (1 - \epsilon_{i,s})^{\frac{1}{2(|A_i|-1)}} \epsilon_{i,s}^{1 - \frac{1}{2(|A_i|-1)}} \equiv \varphi(\epsilon_{i,s}). \end{aligned}$$

One can easily show that $\varphi(\epsilon)$ is a strictly concave function of $\epsilon \in [0, 1]$, satisfying

$$\varphi\left(1 - \frac{1}{|A_i|} \right) = 1 \tag{25}$$

Since $\mathcal{H}_i$ is weakly learnable, $\exists \gamma_i > 0$ such that

$$0 \leq \epsilon_k \leq 1 - \frac{1}{|A_i|} - \gamma_i < 1 - \frac{1}{|A_i|}.$$

Since φ is strictly concave, $\varphi'(\epsilon) > 0 \ \forall \epsilon < 1 - \frac{1}{|A_i|}$. Thus, $\exists \hat{\gamma}_i > 0$ such that $\forall \epsilon_k \leq 1 - \frac{1}{|A_i|} - \gamma_i$,

$$\varphi(\epsilon_k) \leq \varphi\left(1 - \frac{1}{|A_i|} - \gamma_i \right) = 1 - \hat{\gamma}_i < 1.$$

Thus,

$$Z_{i,k} = \varphi(\gamma_{i,k}) \le \varphi(\gamma_i) \le 1 - \hat{\gamma}_i$$

for some $\hat{\gamma}_i > 0$. Thus,

$$d_{i,k+1}(x) = d_{i,1}(x) \prod_{s=1}^{k} Z_{i,k} \le \frac{1}{|S|}(1-\hat{\gamma}_i)^k = \frac{1}{|S|} e^{-\rho_i t}$$

where

$$\rho_i = -\log(1-\hat{\gamma}_i) > 0 \tag{26}$$

from which the conclusion of Lemma A.1 follows. □

By the definition of $\Psi^a_{i,k}$, $\forall x, \forall i, k$,

$$\sum_{a \in A_i} \Psi^a_{i,k}(x) = 0. \tag{27}$$

By (27), $\forall x$, if $\Psi^{y(x)}_k(x) > 0$, then $\exists \underline{a}_i(x) \in A_i$ such that

$$\Psi^{\underline{a}_i(x)}_{i,k}(x) < 0.$$

Lemma A.1 implies

$$\underline{a}_i(x) \ne y_i(x)$$

with probably at least $1 - e^{-\rho_i K}$. Thereofre, $\Psi^{y_i(x)}_{i,K}(x) > 0$ almost surely as $K \to \infty$. If $|A| = 2$, then (27) implies that exactly one label is associated with a positive value of $\Psi^a_{1,K}(x)$. Thus, searching for a with $\Psi^a_{i,K}(x) > 0$ is the right way to identify a correct label. If $|A| > 2$, then $\Psi^a_{i,K}(x) > 0$ does not imply that $a = y_i(x)$. However, if $\Psi^a_{i,K}(x) < 0$, then Lemma A.1 implies that $a \ne y_i(x)$ almost surely, as $K \to \infty$. The idea of τ^K_{Iter} is to eliminate a with $\Psi^a_{i,k}(x) < 0$ in each epoch i, until we have a single element left.

Across Epochs. We construct A_{i+1} by eliminating some elements from A_i and permuting the remaining elements according to π_i. With probability $1 - e^{-\rho_i K}$, the true label survives the elimination from A_i, and is included in A_{i+1}, permuted according to π^x_i.

Since we eliminate at least one element from A_i, the total number of epochs cannot be larger than $|A|$. Let I be the epoch when A_I has two elements so that by the end of epoch I, only a single element of A_I survives. With probability

$$\prod_{i=1}^{I}(1 - e^{-\rho_i K}),$$

the remaining label $y_I(x)$ is the true label. By definition,

$$y_I(x) = \lambda^x_I(y_1(x)) = \lambda^x_I(y(x)).$$

Since y^x_i is the composite function of bijective functions, it is a bijection. We can identify the original label associated with $y_I(x)$ by

$$y(x) = (\lambda^x_I)^{-1}(y_I(x)).$$

Define

$$\underline{\rho} = \frac{1}{2}\min \rho_i > 0.$$

Then, for a sufficiently large K,

$$\begin{aligned}\prod_{i=1}^{I}(1 - e^{-\rho_i K}) &\ge 1 - |A|\, e^{-K \min_i \rho_i} = 1 - e^{-K(\min_i \rho_i + \frac{1}{K}\log|A|)} \\ &\ge 1 - e^{-\underline{\rho} K}.\end{aligned}$$

The final classifier misclassifies $x \in S$ if and only if the correct class $y(x)$ is eliminated in some epoch i, i.e., $\Psi^{y_i(x)} i, K(x) \le 0$. Thus, the probability of making a wrong forecast is

$$\mathbf{P}_{\mathcal{D}}\left(\exists i,\ \Psi^{y_i(x)}_{i,K}(x) \le 0\right). \tag{28}$$

Since $I = |A| - 1$ by Theorem 6.2, (28) is bounded by

$$|A| \max_{i \in \{1,\ldots,I\}} \mathbf{P}_{\mathcal{D}}\left(\Psi^{y_i(x)}_{i,K}(x) \le 0\right) \le |A|\, e^{-\underline{\rho} K}. \tag{29}$$

B PROOF OF THEOREM 6.3

The proof follows the same logic as the proof of Theorem 5.1 on page 98 of [12], originally due to [13], using the same notation whenever possible. We must change the notation to accommodate multiple classes ($|A| \ge 2$). We also note necessary modifications for our Theorem 6.3.

In each epoch, we essentially solve the binary labeling problem and can invoke Theorem 5.1 and 5.5 of [12] without any substantive changes other than scaling the bound. Some minor modifications are needed to apply them to our multiclass setting, as the notions of margin and VC-dimension used in the above Theorem are specific to the binary class problem. Nevertheless, we can consider the "projection" onto binary class problems for every multiclass problem, where we only consider whether a classifier is correct or incorrect.

Preliminaries. Fix epoch i. Recall that

$$\Psi^a_{i,k}(x) = \sum_{s=1}^{k} \alpha_{i,s}\left[|A_i|\,\mathbb{I}(h_{i,s}(x) = a) - 1\right].$$

Define

$$g^a_{i,s}(x) = |A_i|\,\mathbb{I}(h_{i,s}(x) = a) - 1,$$

$$\beta_{i,s} = \frac{\alpha_{i,s}}{\sum_{s'=1}^{k} \alpha_{i,s'}}$$

and

$$\psi^a_{i,k}(x) = \sum_{s=1}^{k} \beta_{i,s} g^a_{i,s}(x).$$

We are calculating the upper bound of the probability of making a wrong decision in epoch i: the probability of eliminating $y_i(x)$ in epoch i, or

$$\mathbf{P}\left(\psi^{y_i(x)}_{i,K}(x) \le 0\right)$$

at the end of epoch i, where K is the duration of epoch i. It is not incorrect that $\psi^a_{i,K}(x) > 0$ for some $a \ne y_i(x)$, as long as $\psi^{y_i(x)}_{i,K}(x) > 0$. We can drop $y \in \{+1, -1\}$ from the statements in [12], focusing on the sign of $\psi^{y_i(x)}_{i,K}(x)$.

Recall that $\mathcal{H}_i$ is the collection of feasible weak classifiers, whose generic element is $h_i : X \to A_i$. Define the set of function from $S \times A_i \to [-1, |A_i| - 1]$ as follows:

$$\mathcal{G}_i = \left\{g^a_i(x) \mid \exists h_i \in \mathcal{H}_i,\ g^a_i(x) = |A_i|\,\mathbb{I}(h_i(x) = a) - 1\right\}.$$

Note that $\psi^a_{i,K}(x)$ is a convex combination of elements in $\mathcal{G}_i$. Fix n and $a \in A_i$, define

$$\mathcal{A}_{i,n} = \left\{f^a_{i,n} : x \mapsto \frac{1}{n}\sum_{j=1}^{n} g^a_{i,j}(x) \mid g^a_{i,1}, \ldots, g^a_{i,n} \in \mathcal{G}_i, a \in A_i\right\}.$$

Given a collection of n *randomly* selected elements from $\mathcal{G}_i$, say $\tilde{g}^a_{i,1}, \ldots, \tilde{g}^a_{i,n}$, where

$$\mathbf{E}\tilde{g}^a_{i,j} = \sum_{s=1}^{K} \beta_{i,s} g^a_{i,s}(x) \qquad \forall j \in \{1, \ldots, n\}.$$

Let

$$\tilde{f}^a_{i,n}(x) = \frac{1}{n} \sum_{j=1}^{n} \tilde{g}^a_{i,j}(x).$$

as a sample average of $\{\tilde{g}^a_{i,1}, \ldots, \tilde{g}^a_{i,n}\}$. Notice that, if $\tilde{g}^a_{i,j}(x)$ is derived from $g^a_{i,s}(x)$ with probability distribution $\{\beta_{i,s}\}_{s=1}^K$, then

$$\mathbf{E}\tilde{f}^a_{i,n}(x) = \sum_{s=1}^{K} \beta_{i,s} g^a_{i,s}(x) = \psi^a_{i,K}(x) \tag{30}$$

Define

$$\beta_{i,n,\theta} = 2e^{-\frac{n\theta^2}{2|A_i|^2}} \text{ and } \epsilon_{i,n} = \sqrt{\frac{\ln(n(n+1)^2|\mathcal{H}_i|^n/\delta)}{2m}}.$$

We can prove the following intermediate results, each of which corresponds to a lemma in [12] stated for the case of $|A| = 2$ and $I = 1$.[7]

$$\mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}(x)} \left[\left| \tilde{f}^{y_i(x)}_{1,n}(x) - f^{y_i(x)}_{1,n}(x) \right| \geq \frac{\theta}{2} \right] \leq \beta_{i,n,\theta}, \text{(Lemma 5.2)}$$

$$\mathbf{P}_{P,\tilde{f}^{y_i(x)}_{i,n}} \left[\left| \tilde{f}^{y_i(x)}_{1,n}(x) - f^{y_i(x)}_{1,n}(x) \right| \geq \frac{\theta}{2} \right] \leq \beta_{i,n,\theta} \text{(Lemma 5.3)}$$

$$\mathbf{P}_{d^*} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] \leq \mathbf{P}_{\mathcal{S}} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] + \epsilon_{i,n} \text{(Lemma 5.4)}$$

For the case where $|\mathcal{H}_i| = \infty$ with VC-dimension d_i, the inequalities are the same, except for the last one (Lemma 5.4) where $\epsilon_{i,n}$ changes to

$$\overline{\epsilon}_{i,n} = \sqrt{\frac{32[\ln(n(n+1)^2) + d_i n \ln(em/d_i) + \ln(8/\delta)]}{m}},$$

which is proved in their Lemma 5.6, where d_i is VC-dimension of $\mathcal{H}_i$.

Restatement of Lemma 5.2. We show $\forall n \geq 1$

$$\mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}} \left[\left| \tilde{f}^{y_i(x)}_{i,n}(x) - \psi^{y_i(x)}_{i,n}(x) \right| \geq \frac{\theta}{2} \right] \leq \beta_{i,n,\theta}.$$

This probability distribution is taken over $\tilde{f}^{y_i(x)}_{i,n}$. Notice that an arbitrary distribution over $\tilde{f}^{y_i(x)}_{i,n}$ can be generated by first choosing an arbitrary distribution of the weights $\{\beta_{i,j}\}_{j=1}^n$. Hence $\mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}}$ corresponds to a choice of $\tilde{f}^{y_i(x)}_{i,n}$ generated according to probability distribution $\{\beta_{i,s}\}_{s=1}^K$:

$$\mathbf{E}\tilde{f}^{y_i(x)}_{i,n}(x) = \mathbf{E}\tilde{g}_{i,j}(y_i(x), x) = \psi^{y_i(x)}_{i,n}(x) \qquad \forall j \in \{1, \ldots, n\}.$$

[7] The number at the end of each line is the corresponding lemma in [12]

Furthermore, since $g_{i,j} \in [-1, |A_i| - 1]$, we have $\frac{1}{|A_i|}(g_{i,j}+1) \in [0, 1]$. Hoeffding's inequality therefore implies

$$\begin{aligned} & \mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}} \left[\left| \tilde{f}^{y_i(x)}_{i,n}(x) - \psi^{y_i(x)}_{i,n}(x) \right| \geq \frac{\theta}{2} \right] \\ = \; & \mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}} \left[\left| \left(\frac{1}{|A_i|} \tilde{f}^{y_i(x)}_{i,n}(x) + 1 \right) - \left(\frac{1}{|A_i|} \psi^{y_i(x)}_{i,n}(x) + 1 \right) \right| \geq \frac{\theta}{2|A_i|} \right] \\ \leq \; & 2e^{-2n(\theta^2/(4|A_i|^2))} = 2e^{-n(\theta^2/(2|A_i|^2))} = \beta_{i,n,\theta} \end{aligned}$$

as desired.

Restatement of Lemma 5.3. We show that the same probability bound obtains when replacing $\mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}}$ with $\mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n},S}$ for an arbitrary probability distribution over $\{(x, y_i(x))\}_{x \in S}$ pairs. When $(x, y_i(x))$ is drawn according to an arbitrary distribution instead of being fixed as in Section B. The argument is identical to Lemma 5.3 in [12]. We include this argument for reference since our notation is slightly different. The argument uses the fact that, for random variables X and Y and any event a defined on them, we have $\mathbf{P}_{X,Y}[a] = \mathbf{E}_X[\mathbf{P}_Y[a|x]]$.

$$\begin{aligned} & \mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n},S} \left[\left| \psi^{y_i(x)}_{i,n}(x) - \tilde{f}^{y_i(x)}_{i,n}(x) \right| \geq \frac{\theta}{2} \right] \\ & = \mathbf{E}_S \left[\mathbf{P}_{\tilde{f}^{y_i(x)}_{i,n}} \left[\left| \psi^{y_i(x)}_{i,n}(x) - \tilde{f}^{y_i(x)}_{i,n}(x) \right| \geq \frac{\theta}{2} \right] \right] \\ & \leq \beta_{i,n,\theta}. \end{aligned} \tag{31}$$

Identical arguments from [12] use this observation to show:

$$\mathbf{P}_{d^*} \left[\psi^{y_i(x)}_{i,n}(x) \leq 0 \right] \leq \mathbf{P}_{d^*,\tilde{f}^{y_i(x)}_{i,n}} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] + \beta_{i,n,\theta}.$$

and

$$\mathbf{P}_{\mathcal{S},\tilde{f}^{y_i(x)}_{i,n}} \left[\tilde{f}^{y_i(x)}_{i,n} \leq \frac{\theta}{2} \right] \leq \mathbf{P}_{\mathcal{S}} \left[\psi^{y_i(x)}_{i,n}(x) \leq \theta \right] + \beta_{i,n,\theta}.$$

Restatement of Lemma 5.4. This part follows immediately from [12]. Leaving the $\epsilon_{i,n}$ constant unchanged, we have with probability $1 - \delta$, a sample S is drawn such that for all $n \geq 1$, $\tilde{f}^{y_i(x)}_{i,n} \in \mathcal{A}_{i,n}$, and $\theta \geq 0$,

$$\mathbf{P}_{d^*} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] \leq \mathbf{P}_{\mathcal{S}} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] + \epsilon_{i,n}$$

Restatement of Lemma 5.6. For the case of $|\mathcal{H}| = \infty$, the same observation applies in the sense that we also obtain the bound

$$\mathbf{P}_{d^*} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] \leq \mathbf{P}_{S} \left[\tilde{f}^{y_i(x)}_{i,n}(x) \leq \frac{\theta}{2} \right] + \overline{\epsilon}_{i,n}.$$

The proof in [12] involves the VC-dimension of the set $\mathcal{H}$ of weak classifiers. In our case, the interest is in the set of binary classifiers *derived* from $\mathcal{H}$ where, given an observation x, labels equal to $y_i(x)$ are assigned +1. All others are assigned −1. This is precisely the graph dimension of $\mathcal{H}$, and hence the same proof carries over, using $\overline{\epsilon}_{i,n}$ involving the graph dimension of $\mathcal{H}$ in place of the VC.