MadaBoost: A Modification of AdaBoost

Carlos Domingo[*]
Dept. of Math. and Comput. Sci.
Tokyo Institute of Technology
carlos@is.titech.ac.jp

Osamu Watanabe[†]
Dept. of Math. and Comput. Sci.
Tokyo Institute of Technology
watanabe@is.titech.ac.jp

Abstract

We propse a new boosting algorithm that mends some of the problems that have been detected in the so far most successful boosting algorithm, **AdaBoost** due to Freund and Schapire [FS97]. These problems are: (1) **AdaBoost** cannot be used in the boosting by filtering framework, and (2) **AdaBoost** does not seem to be noise resistant. In order to solve them, we propose a new boosting algorithm **MadaBoost** by modifying the weighting system of **AdaBoost**. We prove that one version of **MadaBoost** is in fact a boosting algorithm, and we show how our algorithm can be used in detail. We then prove that our new boosting algorithm can be casted in the statistical query learning model [Kea93] and thus, it is robust to random classification noise [AL88].

1 Introduction

In the last decade, boosting techniques have been received a great deal of attention from the machine learning and computational learning communities. In this paper, we further explore this topic by proposing a new boosting algorithm — **MadaBoost** — that mends some of the problems that have been detected in the so far most successful boosting algorithm, **AdaBoost** due to Freund and Schapire [FS97]. These problems are: (1) **AdaBoost** cannot be used in the boosting by filtering framework, and (2) **AdaBoost** does not seem to be noise resistant. The outline of our modification was first proposed in [Wat99] with only a partial proof for its justification. In this paper, we describe the modification in detail, provide a much improved analysis of its correctness and performance, and prove that our new boosting algorithm can be casted in the statistical query learning model [Kea93] and thus, it is robust to random classification noise [AL88].

While the above problem (2) is an obvious weakness, it may not seem so important that even if a boosting algorithm does not work for the "filtering framework", where examples are randomly obtained with respect to a distribution defined over all instance space. Here we argue that it is indeed important that a boosting algorithm can be used in the filtering framework.

Recall that **AdaBoost** is defined for the "subsampling framework", where a sample of sufficient size, which is randomly selected before the boosting, is fixed throughout all the boosting process and distributions are defined only with respect to the sample. From the theoretical side, one consequence of having a boosting algorithm for the filtering framework is that we can directly get a bound on the generalization error of the boosting algorithm. From a more practical side, the advantage becomes more clear. First, we do not need to determine "sufficient sample size" for the boosting process. (Although some formulas exist to calculate sample size, they may not be easy to use and they usually give overestimated size.) Secondly, since a sample is not a priori fixed, we can run the weak learner on random samples of appropriate sizes at each iteration of the boosting; in this way, we can reduce the computation time particularly when the dataset is very large and we use apprporiate sampling [DGW99] for scaling-up the weak learner. Recall that **AdaBoost** is defined for the subsampling framework and that it is not appropriate for the filtering framework, at least not in an obvious way.

Note that there are boosting algorithms, namely, the one proposed by Schapire [Sch90] and the one by Freund [Fre95], that can be used for the filtering framework and that are noise resistant [AD93] in the sense of the statistical query learning model. However, none of these algorithms are "adaptive" like **AdaBoost**, a property that has been repeatedly shown to be crucial for practical feasibility. More recently, some attempts have been done for improving **AdaBoost**, mainly to try to make it noise resistant. These modifications also tried to change the weighting scheme so that the weights change more smoothly; see, e.g., [FHT98]. From these variations, the only one that is a boosting algorithm in the PAC sense is BrownBoost [Fre99]. While BrownBoost seems more error resistant than **AdaBoost**, it has not been shown whether it can be used in the filtering framework. Moreover, BrownBoost is a more complicated than **AdaBoost**, while our algorithm remains as simple as **AdaBoost**.

Our modification for **MadaBoost** is very simple. We just bound the weight assigned to each example by its initial probability. In this way, the weights of the examples cannot become arbitrarily large as it happens in **AdaBoost**. The uncontrolled growth of the weights seems to be the root

[*]Supported in part by European Union Science and Technology Fellowship, STF13.

[†]Supported in part by the Ministry of Education, Science, Sports and Culture of Japan, Grant-in-Aid for Scientific Research on Priority Areas (Discovery Science).

of the problems of **AdaBoost**, and in fact, we can show that **MadaBoost** works under the filtering framework and it belongs to the statistical query model. What is more important from the theoretical view point is that we can show that even this moderate weight scheme has the boosting property. Unfortunately, our proof still has some minor weaknesses. First, the proof works only for one version of **MadaBoost** that has an even more moderate weight scheme. Secondly, we need to assume that the advantages of the weak hypotheses produced during the boosting process are monotonically decreasing. Thirdly, the boosting speed, at least what we could prove, is much slower than the one for **AdaBoost**. Nevertheless, we do not expect these weaknesses to affect the practicality of our algorithm, an intuition supported by our experiments [DW00].

2 Preliminaries

We explain the notion of boosting and **AdaBoost**. (Please do not skip our description of **AdaBoost** that is different from the original one.)

We begin by recalling some basic notions on PAC learning. The goal of PAC learning is to obtain some hypothesis f that can predict an unknown target function f_* with a desired accuracy under an unknown distribution D on the domain X of f_*. Throughout this paper, we consider the problem of learning Boolean functions. Let $\mathbf{EX}_{D,f_*}$ be a black box example generator, which generates a pair $(x, f_*(x))$ with probability $D(x)$. An algorithm A is called a *PAC learner* if for any D, and for any input ϵ and δ, $0 < \epsilon, \delta < 1$, by using $\mathbf{EX}_{D,f_*}$, A yields some hypothesis f such that $\mathrm{cor}_D(f, f_*)$ ($\stackrel{\mathrm{def}}{=} \Pr_{x:D}\{ f(x) = f_*(x) \}$) $\geq 1-\epsilon$ with probability at least $1-\delta$. On the other hand, an algorithm is called a *weak PAC learner* if for any D and any input δ, $0 < \delta < 1$, it uses $\mathbf{EX}_{D,f_*}$ and obtains, with probability at least $1-\delta$, a hypothesis h such that $\mathrm{cor}_D(h, f_*) = 1/2+\gamma$ for some $\gamma > 0$. This γ is called the *advantage* of h (over the random hypothesis). The efficiency (e.g., the running time) of a learning algorithm is usually measured in terms of $1/\epsilon$ (or $1/\gamma$ in case of a weak learner), $1/\delta$, the size of instances, and the "complexity" of a target function. In order to simplify our discussion, we will ignore instance size and complexity of the target function, and we measure the efficiency only in terms of $1/\delta$ and $1/\epsilon$ and/or $1/\gamma$.

Boosting Techniques: Subsampling and Filtering

We explain the outline of boosting techniques. Suppose that we are given some weak learning algorithm **WeakLearn**. For the rest of the paper we will assume that **WeakLearn** is an algorithm that, when given as input a confidence value δ, it asks for a certain amount of labeled examples under a fixed but unknown distribution and, with probability larger than $1-\delta$, outputs a weak hypothesis that has error strictly smaller than 1/2 under the distribution from which the examples were generated.

A boosting algorithm runs this **WeakLearn** several times, say T times, under distributions $D_1, ..., D_T$ that are slightly modified from the given distribution D and collects weak hypotheses $h_1, ..., h_T$. (In the following, we call each execution of **WeakLearn** a *boosting round* or *boosting step*.) A final hypothesis is built by combining these weak hypotheses. Here the key idea is to put more weight, when making a new weak hypothesis, to "problematic instances" for which the previous weak hypotheses perform poorly. That is, at the point when $h_1, ..., h_{t-1}$ have been obtained, the boosting algorithm computes a new distribution D_t that puts more weight on those instances that have been misclassified by most of $h_1, ..., h_{t-1}$. Then, a new hypothesis h_t produced by **WeakLearn** on this distribution D_t should be strong on those problematic instances, thereby improving the performance of the combined hypothesis built from $h_1, ..., h_t$.

Boosting techniques differ typically on (i) the weighting scheme used to obtained the modified distribution, (ii) the way the weak hypotheses are combined, and (iii) the way to execute **WeakLearn** on the modified distributions. According to this last point, boosting techniques can be classified in two types:- *boosting by subsampling* and *boosting by filtering* [Fre95]. In the boosting by subsampling framework, a boosting algorithm first obtains, using $\mathbf{EX}_{D,f_*}$, a set S of enough number of examples as a "training set". Then it runs **WeakLearn** on S by changing the weight of each example. The goal is to obtain a hypothesis that explains the training set well and hope that this hypothesis will also predict well the label on unseen examples outside S. Since we can now regard S as a domain, we only need to provide labeled examples under a given distribution on S, which is possible so long as the distribution is efficiently computable. On the other hand, in the boosting by filtering framework, a boosting algorithm selects an example from the original domain X each time **WeakLearn** requests one.

AdaBoost: An Adaptive Boosting Algorithm

Here we describe **AdaBoost** in detail. **AdaBoost** is designed for the subsampling framework. Let S be the sample that is given as input to **AdaBoost**. For any $t \geq 1$, assume that we have already obtained hypotheses $h_1, ..., h_{t-1}$, where each h_i is a weak hypothesis of f_* on some distribution D_{i-1} defined with respect to sample S. Let $\epsilon_1, ..., \epsilon_{t-1}$ denote the errors and $\gamma_1, ..., \gamma_{t-1}$ denote the advantages of these hypotheses; that is, $\epsilon_i \stackrel{\mathrm{def}}{=} 1 - \mathrm{cor}_{D_i}(h_i)$ and $\gamma_i \stackrel{\mathrm{def}}{=} \mathrm{adv}_{D_i}(h_i)$ ($= 1/2 - \epsilon_i$). We use parameters $\beta_1, ..., \beta_{t-1}$ that are defined as $\beta_i \stackrel{\mathrm{def}}{=} \sqrt{\epsilon_i/(1-\epsilon_i)}$ ($= \sqrt{(1-2\gamma_i)/(1+2\gamma_i)}$) for each i, $1 \leq i \leq t-1$. (*Remark.* In this paper, we define β_i in this way, which is the square root of β_i used in [FS97]. Due to this change, the weight w_t defined below is not the same as the one used in [FS97] although the definitions of the distributions D_t and the combined hypothesis f_t also given below are mathematically equivalent. Thus, there is no essential difference between this description of the algorithm and the original one, while ours is more appropriate for discussing the modification to the filtering framework.) For any hypothesis h and any $x \in X$, define $\mathrm{cons}(h, x) \stackrel{\mathrm{def}}{=} 1$ (resp., -1) if $h(x) = f_*(x)$ (resp., $h(x) \neq f_*(x)$). Then for each instance $x \in S$, its weight $w_{t-1}(x)$ after the $(t-1)$th boosting round is defined as follows.

$$w_{t-1}(x) \stackrel{\mathrm{def}}{=} D_0(x) \times \prod_{1 \leq i \leq t-1} \beta_i^{\mathrm{cons}(h_i, x)},$$

where D_0 is the initial distribution, that is the uniform distribution on S. Let $W_{t-1} \stackrel{\mathrm{def}}{=} \sum_{x \in S} w_{t-1}(x)$. The next dis-

tribution D_t is defined as $D_t(x) \stackrel{\text{def}}{=} w_{t-1}(x)/W_{t-1}$ for all $x \in S$. (Note that $D_1(x) = D_0(x)$.) Finally, the combined hypothesis f_{t-1} of $h_1, ..., h_{t-1}$ is their weighted majority vote that is defined as

$$f_{t-1}(x) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } \prod_{i:h_i(x)=1} \beta_i \le \prod_{i:h_i(x)=0} \beta_i, \\ 0, & \text{otherwise.} \end{cases}$$

Then for the "boosting property" of **AdaBoost**, we have the following theorem [FS97].

Theorem 1 *Suppose that* **WeakLearn***, when called by* **AdaBoost** *on a training set S, generates weak hypotheses $h_1, h_2, ..., h_T$ whose advantages are $\gamma_1, \gamma_2, ..., \gamma_T > 0$. Then the error of the combined hypothesis f_T on S can be bounded as follows.*

$$\text{error}(S, f_T) \stackrel{\text{def}}{=} \Pr_{x:D_0}\{ f_T(x) \neq f_*(x) \} \le \exp\left(-2\sum_{i=1}^{T} \gamma_i^2\right).$$

3 MadaBoost: A New Weighting Scheme

Here we describe our modification for our boosting algorithm **MadaBoost** and prove that one version of **MadaBoost** has an adaptive boosting property. For the sake of comparison we describe **MadaBoost** first in the subsampling framework. Later we will explain how to make it work for the filtering framework.

The modification we propose is very simple. For each instance x, we modify its weight $w_t(x)$ after the tth boosting round with the initial weight $D_0(x)$ as a saturation bound; that is, the weight $w_t(x)$ cannot be increased beyond $D_0(x)$. (Recall that D_0, in the subsampling framework, is usually taken as the uniform distribution over S.) More specifically, our new weighting system is defined as follows. Define $B_t(x) \stackrel{\text{def}}{=} \prod_{1 \le i \le t} \beta_i^{\text{cons}(h_i,x)}$. Then for each instance $x \in S$, the new weight $w_{t-1}(x)$ after the $(t-1)$th step is defined by

$$w_{t-1}(x) \stackrel{\text{def}}{=} \begin{cases} D_0(x) \times B_{t-1}(x), & \text{if } B_{t-1}(x) < 1, \text{ and} \\ D_0(x), & \text{otherwise.} \end{cases}$$

The rest is exactly the same as before. Recall that $W_{t-1} \stackrel{\text{def}}{=} \sum_{x \in S} w_{t-1}(x)$ and $D_t(x) \stackrel{\text{def}}{=} w_{t-1}(x)/W_{t-1}$ for all $x \in S$.

Thus, the weight of each instance changes very moderately. It is interesting to see that even using this moderate weighting scheme the algorithm still has some boosting property. For example, our experiments [DW99] show that **MadaBoost** has a boosting property more or less similar to **AdaBoost**. Here we prove a general adaptive boosting property. Unfortunately, though, for our current proof, we need to modify the weighting scheme of **MadaBoost** even more moderate one. The difference from **MadaBoost** is the definition of β_t; we define $\beta_t = \sqrt{\epsilon'_t/(1-\epsilon'_t)}$, where $\epsilon'_t = \sqrt{\epsilon_t/2}$. That is, instead of using the error probability ϵ_t of the tth weak hypothesis, we use ϵ'_t, which is slightly larger than ϵ_t. In terms of the advantage γ_t, since $\epsilon'_t \approx 1/2 - \gamma_t/2$, this new weighting scheme defines β_t by using the advantage that is roughly the half of the real one. Note also that $\epsilon'_t < 1/2$ so long as $\epsilon_t < 1/2$. We refer this version of **MadaBoost** as **MB:1/2**. For this algorithm, we can prove the following boosting property.

Theorem 2 *Suppose that* **WeakLearn***, when called by* **MB:1/2** *on a training set S, generates weak hypotheses $h_1, h_2, ..., h_T$ whose advantages are $\gamma_1, \gamma_2, ..., \gamma_T$ that satisfies $\gamma_1 \ge \gamma_2 \ge \cdots \ge \gamma_T > 0$. Let $f_1, f_2, ..., f_T$ be combined hypotheses obtained after each round. Then for any ϵ, either there is some t, $1 \le t < T$, for which we have*

$$\text{error}(S, f_t) = \Pr_{x:D_0}\{ f_t(x) \neq f_*(x) \} < \epsilon,$$

or we have the following bound.

$$\text{error}(S, f_T) = \Pr_{x:D_0}\{ f_T(x) \neq f_*(x) \} \le 1 - \sum_{1 \le i \le T} 2\epsilon\gamma_i^2.$$

Before the proof, let us examine the meaning of this theorem. Suppose that our goal is to obtain a hypothesis whose error probability on S is less than $\epsilon = 1/\|S\|$ (under the uniform distribution over S), and suppose that **WeakLearn** generates weak hypotheses with advantage larger than γ. Then within $T = (\|S\| - 1)/(2\gamma^2)$ rounds, we have some f_t with $\text{error}(S, f_t) < \epsilon$ because if no f_t, $1 \le t < T$, has this desired property, then we have $\text{error}(S, f_T) \le 1 - T \cdot (2\gamma^2/\|S\|) \le 1/\|S\|$; that is, f_T indeed has the desired property. Hence, we can bound the number of boosting rounds as $\mathcal{O}(\|S\|/\gamma^2)$, or $\mathcal{O}(\epsilon^{-1}/\gamma^2)$ in general, to reduce the error in S below ϵ. On the other hand, **AdaBoost** needs $\mathcal{O}(\ln \epsilon^{-1}/\gamma^2)$ under the same situation. That is, the boosting speed of **MB:1/2**, at least the one we can prove here, is exponentially slower than **AdaBoost** in terms of ϵ^{-1}.

Here we need one condition; that is, the advantage sequence $\gamma_1, \gamma_2, \ldots, \gamma_T$ is non-increasing. Intuitively, a learning problem gets harder and harder as boosting proceeds; hence, it seems natural to assume that an advantage sequence is non-increasing. In fact, this phenomenon has been confirmed experimentally [DKM96, DW99]. Note also that, even if it occurs $\gamma_t > \gamma_{t-1}$ for some round t, we can continue boosting by using γ_{t-1} instead of γ_t. That is, **MB:1/2** works with no problem with any advantage sequence; it just cannot use the advantage of some "accidentally" good weak hypothesis.

For proving the theorem, we first note the following fact, which will be also important for our later discussion. (The same property holds for **MadaBoost** and **AdaBoost**.)

Lemma 3 *Suppose that* **MB:1/2** *is executed with a sample set S in which it executes* **WeakLearn** *for t times and obtains weak hypotheses $h_1, ..., h_t$ with advantage $\gamma_1, ..., \gamma_t$. Let f_t be the combined hypothesis obtained from them. Then we have* $\text{error}(S, f_t) \le W_t$.

Therefore, in order to prove Theorem 2, it suffices to show that the sequence $W_1, W_2, ...$ converges to 0. More specifically, we need to show that W_t is smaller than W_{t-1} by $(\epsilon/2)\gamma_t^2$ (if $W_{t-1} \ge \epsilon$). This is what we will prove below.

Proof of Theorem 2. For any $t \ge 1$, we consider the tth boosting step, the situation when the tth boosting round has just finished. That is, **WeakLearn** has been called for t times and t weak hypotheses $h_1, \ldots, h_t$ have been obtained. Let $\epsilon_1, \ldots, \epsilon_t$ be their errors, and let $\gamma_1, \ldots, \gamma_t$ be their advantages. We would like to discuss how much W_t gets decreased

from W_{t-1}, but it seems difficult to estimate the decrement because there may be some instance $x \in X$ for which the weight does not change between the $(t-1)$th and the tth step. Thus, we introduce here some "imaginary" weight that bounds W_t and show that it gets decreased as the theorem claims.

First we define our new weight for the situation that the $(t-1)$th step has been finished. For this, we divide the instance space X according to the value $B_{t-1}(x) = \prod_{i=1}^{t-1} \beta_i^{\mathrm{cons}(h_i,x)}$ of each instance $x \in X$. More specifically, we divide X into two sets U and V, where U (resp., V) is the set of instances $x \in X$ such that $B_{t-1}(x) < 1$ (resp., $B_{t-1}(x) \geq 1$). Note that x is in U iff f_{t-1} gives a correct classification, and note that for any $x \in V$, its weight $w_{t-1}(x)$ is defined by $w_{t-1}(x) = D_0(x)$. Now by using $\alpha_t \stackrel{\mathrm{def}}{=} \beta_t^{-1} - 1 > 0$, we define our new weight $\widetilde{w}_{t-1}(x)$ as follows.

$$\begin{array}{ll} (\text{if } x \in U) & \\ \quad \widetilde{w}_{t-1}(x) & \stackrel{\mathrm{def}}{=} \quad B_{t-1}(x)D_0(x), \text{ and} \\ (\text{if } x \in V) & \\ \quad \widetilde{w}_{t-1}(x) & \stackrel{\mathrm{def}}{=} \quad (1 + \alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x). \end{array}$$

Also define $\widetilde{W}_{t-1} \stackrel{\mathrm{def}}{=} \sum_{x \in X} \widetilde{w}_{t-1}(x)$.

We explain the motivation of this weight function. In our original weight scheme, all instances in U changes their weight from the $(t-1)$th step to the tth step, while some instances x in V keep the same weight, i.e., $D_0(x)$. Thus, for the weight $\widetilde{w}_{t-1}(x)$ for each $x \in V$, we introduce the additional *nonnegative* term $\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x)D_0(x)$ so that $\widetilde{w}_{t-1}(x)$ changes depending whether h_t gives the correct classification of x or not. Intuitively, if h_t correctly (resp., wrongly) classifies $x \in V$, then its weight gets decreased (resp., increased) by α_t.

We need to be a bit careful for defining the new weight for the tth step. The weight $\widetilde{w}_t$ after the tth step is defined similarly, but the division of X is different and β_{t+1} should be used here. That is, these are defined as follows by using $B_t(x) = \prod_{i=1}^{t} \beta_i^{\mathrm{cons}(h_i,x)}$ and $\alpha_{t+1} = \beta_{t+1}^{-1} - 1$.

$$\begin{array}{rcl} U' & \stackrel{\mathrm{def}}{=} & \{\, x \in X \mid B_t(x) < 1 \,\}, \text{ and} \\ V' & \stackrel{\mathrm{def}}{=} & \{\, x \in X \mid B_t(x) \geq 1 \,\}. \end{array}$$

$$\begin{array}{ll} (\text{if } x \in U') & \\ \quad \widetilde{w}_t(x) & \stackrel{\mathrm{def}}{=} \quad B_t(x)D_0(x), \text{ and} \\ (\text{if } x \in V') & \\ \quad \widetilde{w}_t(x) & \stackrel{\mathrm{def}}{=} \quad (1 + \alpha_{t+1} \log_{\beta_{t+1}^{-1}} B_t(x))D_0(x). \end{array}$$

Then define $\widetilde{W}_t \stackrel{\mathrm{def}}{=} \sum_{x \in X} \widetilde{w}_t(x)$. On the other hand, for our comparison, we also consider an intermediate weight function $\widetilde{w}'_t$ that is defined as follows.

$$\begin{array}{ll} (\text{if } x \in U') & \\ \quad \widetilde{w}'_t(x) & \stackrel{\mathrm{def}}{=} \quad B_t(x)D_0(x), \\ (\text{if } x \in V') & \\ \quad \widetilde{w}'_t(x) & \stackrel{\mathrm{def}}{=} \quad (1 + \alpha_t \log_{\beta_t^{-1}} B_t(x))D_0(x). \end{array}$$

Define $\widetilde{W}'_t \stackrel{\mathrm{def}}{=} \sum_{x \in X} \widetilde{w}'_t(x)$.

Below we will show that $\widetilde{W}_t$ gets decreased from $\widetilde{W}_{t-1}$ by $(\gamma_t^2/2)W_{t-1}$ by proving that (i) $\widetilde{W}_t \leq \widetilde{W}'_t$, and (ii) $\widetilde{W}'_t \leq \widetilde{W}_{t-1} - (\gamma_t^2/2)W_{t-1}$.

First prove $\widetilde{W}_t \leq \widetilde{W}'_t$. For this, it suffices to show that $\widetilde{w}_t(x) \leq \widetilde{w}'_t(x)$ for all $x \in X$. The case that $x \in U'$ is trivial because by definition $\widetilde{w}_t(x) = \widetilde{w}'_t(x)$. Consider the case $x \in V'$. Here we note that $\beta_{t+1}^{-1} \leq \beta_t^{-1}$; this is because we assumed that $\gamma_{t+1} \leq \gamma_t$. Thus we have

$$\begin{array}{rcl} \widetilde{w}_t(x) \;\leq\; \widetilde{w}'_t(x) & & \\ \Leftrightarrow & \alpha_{t+1} \log_{\beta_{t+1}^{-1}} B_t(x) \;\leq\; \alpha_t \log_{\beta_t^{-1}} B_t(x) & \\ \Leftrightarrow & (\beta_t^{-1} - 1)(\log_2 B_t(x)/\log_2 \beta_t^{-1}) & \\ & \quad \leq\; (\beta_{t+1}^{-1} - 1)(\log_2 B_t(x)/\log_2 \beta_{t+1}^{-1}) & \\ \Leftrightarrow & (\beta_t^{-1} - 1)/\log_2 \beta_t^{-1} \;\leq\; (\beta_{t+1}^{-1} - 1)/\log_2 \beta_{t+1}^{-1}. & \end{array}$$

This last inequality holds because $\phi(z) = (z-1)/\log_2 z$ is monotonically incrasing for $z > 1$.

Next we analyze how much does $\widetilde{W}'_t$ get decreased from $\widetilde{W}_{t-1}$. For this, we estimate $\Delta(x) = \widetilde{w}'_t(x) - \widetilde{w}_{t-1}(x)$ considering the cases (U) $x \in U$ and (V) $x \in V$. Furthermore, for each case, we consider the case (P) h_t correctly classifies x and the case (Q) h_t wrongly classifies x. Below we use P and Q to denote $\{\, x \in X \mid h_t(x) = f_*(x) \,\}$ and $\{\, x \in X \mid h_t(x) \neq f_*(x) \,\}$ respectively.

(Case U.P) Note that $x \in U'$. Hence, by definition, we have

$$\begin{array}{rcl} \widetilde{w}_{t-1}(x) & = & B_{t-1}(x)D_0(x) \;=\; w_{t-1}(x), \text{ and} \\ \widetilde{w}'_t(x) & = & B_t(x)D_0(x) \;=\; \beta_t w_{t-1}(x). \end{array}$$

Thus $\Delta(x) = (\beta_t - 1)w_{t-1}(x)$ (≤ 0).

(Case U.Q) We have either $x \in U'$ or $x \in V'$. For the former case, we have

$$\begin{array}{rcl} \widetilde{w}_{t-1}(x) & = & B_{t-1}(x)D_0(x) \;=\; w_{t-1}(x), \text{ and} \\ \widetilde{w}'_t(x) & = & B_t(x)D_0(x) \;=\; \beta_t^{-1} w_{t-1}(x), \end{array}$$

and hence $\Delta(x) = (\beta_t^{-1} - 1)w_{t-1}(x)$ (≥ 0). For the latter case, we have

$$\begin{array}{rcl} \widetilde{w}_{t-1}(x) & = & B_{t-1}(x)D_0(x), \text{ and} \\ \widetilde{w}'_t(x) & = & (1 + \alpha_t \log_{\beta_t^{-1}} B_t(x))D_0(x) \\ & = & (1 + \alpha_t + \alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x). \end{array}$$

Thus, we have

$$\begin{array}{l} \Delta(x) \;=\; (1 - B_{t-1}(x) + \alpha_t + \alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x) \\ \;=\; (\alpha_t + (1 - B_{t-1}(x) + \alpha_t \log_{\beta_t^{-1}} B_{t-1}(x)))D_0(x). \end{array}$$

Here we want to show that $1 - B_{t-1}(x) + \alpha_t \log_{\beta_t^{-1}} B_{t-1}(x) \leq 0$. Note that $\beta_t \leq B_{t-1}(x) < 1$ because $x \in U$ (i.e., $B_{t-1}(x) < 1$) and $x \in V'$ (i.e., $B_t(x) = B_{t-1}(x)\beta_t^{-1} \geq 1$). Hence, $B_{t-1}(x) = \beta_t^z$ for some z, $0 < z \leq 1$. Then we have

$$1 - B_{t-1}(x) + \alpha_t \log_{\beta_t^{-1}} B_{t-1}(x) \quad = \quad 1 - \beta_t^z - \alpha_t z$$

On the other hand, the function $\psi(z) \stackrel{\mathrm{def}}{=} 1 - \beta_t^z - \alpha_t z$ is nonincreasing on $[0,1]$ (since $\alpha_t = \beta_t^{-1} - 1 \geq -\ln \beta_t$) and $\psi(0) = 0$. Thus, we can show that

$$\begin{array}{rcl} \Delta(x) & \leq & \alpha_t D_0(x) \;\leq\; \beta_t^{-1} \alpha_t B_{t-1}(x) D_0(x) \\ & = & \beta_t^{-1} \alpha_t w_{t-1}(x). \end{array}$$

(Case V.P) We have either $x \in U'$ or $x \in V'$. For the latter case, we have

$$\begin{aligned}\widetilde{w}_{t-1}(x) &= (1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x), \text{ and}\\ \widetilde{w}'_t(x) &= (1+\alpha_t \log_{\beta_t^{-1}} B_t(x))D_0(x)\\ &= (1-\alpha_t+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x).\end{aligned}$$

Hence $\Delta(x) = -\alpha_t D_0(x) = -\alpha_t w_{t-1}(x)$ (≤ 0). For the former case, we have

$$\begin{aligned}\widetilde{w}_{t-1}(x) &= (1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x), \text{ and}\\ \widetilde{w}'_t(x) &= B_t(x)D_0(x) = B_{t-1}(x)\beta_t D_0(x)\end{aligned}$$

Let us consider here backwards. For our later analysis, we would like to have $\Delta(x) \leq (\beta_t - 1)w_{t-1}(x)$. But for this, it is enough to have $\Delta(x) \leq (\beta_t - 1)\widetilde{w}_{t-1}(x)$ since $B_{t-1}(x) \geq 1$ and thus $\widetilde{w}_{t-1}(x) \geq w_{t-1}(x)$. On the other hand, since

$$\Delta(x) = (B_{t-1}(x)\beta_t - (1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x)))D_0(x),$$

we have

$$\begin{aligned}&\Delta(x) \leq (\beta_t - 1)\widetilde{w}_{t-1}(x)\\ \Leftrightarrow\ & B_{t-1}(x)\beta_t - (1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))\\ &\quad \leq (\beta_t - 1)(1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))\\ \Leftrightarrow\ & B_{t-1}(x) \leq 1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x)\\ \Leftrightarrow\ & \frac{B_{t-1}(x)-1}{\log_{\beta_t^{-1}} B_{t-1}(x)} \leq \alpha_t.\end{aligned}$$

To prove the last inequality, we note that $1 \leq B_{t-1}(x) < \beta_t^{-1}$. Thus, by letting $\beta_t^{-1} = 1+\delta$ and $B_{t-1}(x) = (1+\delta)^z$ for some $\delta, z > 0$, we have

$$\frac{B_{t-1}(x)-1}{\log_{\beta_t^{-1}} B_{t-1}(x)} = \frac{(1+\delta)^z - 1}{z} \leq \frac{z\delta}{z} = \delta = \beta_t^{-1} - 1.$$

Thus, the desired inequality holds since we defined $\alpha_t = \beta_t^{-1} - 1$. In fact, this is one of the reasons for our definition of α_t.

(Case V.Q) Since $x \in V'$, we have

$$\begin{aligned}\widetilde{w}_{t-1}(x) &= (1+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x), \text{ and}\\ \widetilde{w}'_t(x) &= (1+\alpha_t \log_{\beta_t^{-1}} B_t(x))D_0(x)\\ &= (1+\alpha_t+\alpha_t \log_{\beta_t^{-1}} B_{t-1}(x))D_0(x).\end{aligned}$$

Hence $\Delta(x) = \alpha_t D_0(x) = (\beta_t^{-1} - 1)w_{t-1}(x)$ (≥ 0).

Summarize our analysis. Note that $\alpha_t = \beta_t^{-1} - 1 \geq 1 - \beta_t$. Then for any $x \in X$, we have shown that

$$\begin{aligned}&x \in P\\ &\Rightarrow \Delta(x) \leq \max(\beta_t - 1, -\alpha_t)w_{t-1}(x)\\ &\qquad \leq (\beta_t - 1)w_{t-1}(x), \text{ and}\\ &x \in Q\\ &\Rightarrow \Delta(x) \leq \max(\beta_t^{-1}\alpha_t, \beta_t^{-1} - 1)w_{t-1}(x)\\ &\qquad \leq \beta_t^{-1}(\beta_t^{-1} - 1)w_{t-1}(x).\end{aligned}$$

Thus, the total difference $\Delta \stackrel{\text{def}}{=} \widetilde{W}'_t - \widetilde{W}_{t-1}$ is estimated as follows.

$$\begin{aligned}\Delta &= \sum_{x\in X}\Delta(x) = \sum_{x\in P}\Delta(x) + \sum_{x\in Q}\Delta(x)\\ &\leq \sum_{x\in P}(\beta_t - 1)w_{t-1}(x) + \sum_{x\in Q}\beta_t^{-1}(\beta_t^{-1} - 1)w_{t-1}(x)\\ &= W_{t-1} \times ((\beta_t - 1)(1-\epsilon_t) + \beta_t^{-1}(\beta_t^{-1} - 1)\epsilon_t).\end{aligned}$$

Where the last equality is from the definition of P, Q, and ϵ_t; that is, $\epsilon_t = \Pr_{x:D_{t-1}}\{ h_t(x) \neq f_*(x) \} = D_{t-1}(Q)$, and $1 - \epsilon_t = \Pr_{x:D_{t-1}}\{ h_t(x) = f_*(x) \} = D_{t-1}(P)$.

Here recall that β_t is defined as $\sqrt{\epsilon'_t/(1-\epsilon'_t)}$ with $\epsilon'_t = \sqrt{\epsilon_t/2}$. Hence we have $\epsilon_t \leq \epsilon'_t$ and

$$\beta_t^{-1}\epsilon_t = 2(\epsilon'_t)^2\sqrt{\frac{1-\epsilon'_t}{\epsilon'_t}} = \epsilon'_t(2\sqrt{\epsilon'_t(1-\epsilon'_t)}) \leq \epsilon'_t.$$

Then we can bound Δ as follows.

$$\begin{aligned}\Delta &= W_{t-1} \times ((\beta_t - 1)(1-\epsilon_t) + \beta_t^{-1}(\beta_t^{-1} - 1)\epsilon_t)\\ &= W_{t-1} \times ((\beta_t - 1)(1-\epsilon_t) + \beta_t^{-2}\epsilon_t - \beta_t^{-1}\epsilon_t)\\ &\leq W_{t-1} \times ((\beta_t - 1)(1-\epsilon_t) + \beta_t^{-1}\epsilon'_t - \beta_t^{-1}\epsilon_t)\\ &= W_{t-1} \times (\beta^{-1}\epsilon'_t + (1-\epsilon'_t)\beta_t - 1\\ &\qquad +(\epsilon'_t\beta_t - \epsilon_t\beta_t + \epsilon_t - \epsilon_t\beta_t^{-1}))\\ &\leq W_{t-1} \times (\beta^{-1}\epsilon'_t + (1-\epsilon'_t)\beta_t - 1\\ &\qquad +(2\epsilon'_t\beta_t - 2(\epsilon'_t)^2(\beta_t + \beta_t^{-1})))\\ &= W_{t-1} \times (\beta^{-1}\epsilon'_t + (1-\epsilon'_t)\beta_t - 1)\\ &= W_{t-1} \times (2\sqrt{\epsilon'_t(1-\epsilon'_t)} - 1)\\ &= W_{t-1} \times (\sqrt{1-(1-2\epsilon'_t)^2} - 1)\\ &\leq W_{t-1} \times \frac{-(1-2\epsilon'_t)^2}{2}\\ &= W_{t-1} \times \frac{-1-(1-2\gamma_t)+2\sqrt{1-2\gamma_t}}{2}\\ &\leq W_{t-1} \times \frac{(-2+2\gamma_t)+(2-2\gamma_t-\gamma_t^2)}{2}\\ &= W_{t-1} \times \left(-\frac{\gamma_t^2}{2}\right).\end{aligned}$$

That is, $\widetilde{W}'_t$ (hence $\widetilde{W}_t$) gets decreased from $\widetilde{W}_{t-1}$ at least by $W_{t-1}(\gamma_t^2/2)$.

Now suppose that W_{t-1} is bigger than ϵ. (Otherwise we are done.) Then $\widetilde{W}_t$ decreases from $\widetilde{W}_{t-1}$ by at least $(\epsilon/2)\gamma_t^2$. On the other hand, $\widetilde{W}_0 = W_0 = 1$. Therefore, if $W_t \geq \epsilon$ for all t, $1 \leq t < T$, then we have $W_T \leq \widetilde{W}_T \leq 1 - (\epsilon/2)\sum_{1\leq i\leq T}\gamma_i^2$, as claimed. □

Two remarks from the above proof. First consider the reason why the further modification of **MadaBoost** is necessary. This is because we estimated $\Delta(x)$ somewhat larger. More specifically, for the case $x \in Q$, we estimated $\Delta(x)$ $\leq \max(\beta_t^{-1}\alpha_t, \beta_t^{-1} - 1)w_{t-1}(x) \leq \beta_t^{-1}(\beta_t^{-1} - 1)w_{t-1}(x)$; but we usually would be able to use a smaller bound such as $\Delta(x) \leq (\beta_t^{-1} - 1)w_{t-1}(x)$. Suppose that this bound worked for all (or almost all) $x \in Q$, then we would be able to bound Δ by $W_{t-1} \times (\beta^{-1}\epsilon_t + (1-\epsilon_t)\beta_t - 1) = W_{t-1} \times (2\sqrt{\epsilon_t(1-\epsilon_t)} - 1)$, which gives $\Delta \leq W_{t-1} \times (\sqrt{1-4\gamma_t^2} - 1) \leq W_{t-1} \times (-2\gamma_t^2)$. Then a similar boosting property would be provable for **MadaBoost**.

Next consider the possibility of proving a faster convergence like **AdaBoost**. The reason of our slower convergence is due to the additive decrement of $\widetilde{W}_t$, whereas the weight W_t gets decreased multiplicatively in **AdaBoost**. Note, however, $\widetilde{W}_t$ gets decreased multiplicatively if we may assume that $\widetilde{W}_t \approx W_t$. In fact, assuming that $\widetilde{W}_t \approx W_t$, we would have Δ ($\stackrel{\text{def}}{=} \widetilde{W}_t - \widetilde{W}_{t-1}$) $\leq \widetilde{W}_{t-1} \times (-\gamma_t^2/2)$. Thus, $\widetilde{W}_t \leq$

$\widetilde{W}_{t-1} \times (1 - \gamma_t^2/2)$; that is, $\widetilde{W}_t$ gets decreased multiplicatively and indeed by a factor similar to **AdaBoost**. Since we may assume $\widetilde{W}_t \approx W_t$ for the first several boosting steps, we can expect convergence speed similar to **AdaBoost**, at least, for the first several steps. This phenomenon has been observed in our experiments [DW99].

4 Using MadaBoost with Filtering

We show that **MadaBoost** indeed works under the filtering framework, thereby satisfying our goal. (For simplicity, we use **MadaBoost** for our explanation here and in the next section. The same analysis also holds for **MB:1/2**.) Note that the boosting property of Theorem 2 holds in the filtering framework. The difference is that we consider the error probability on the whole domain X; hence, $D_0 = D$ (in the definition of $w_{t-1}(x)$) and $W_{t-1} = \sum_{x \in X} w_{t-1}(x)$. More specifically, Lemma 3 holds in the filtering framework, and the error probability of the combined hypothesis (under D over X) is bounded by W_t. The main issue now is the way to generate examples under modified distributions.

In the boosting by filtering framework, a boosting algorithm uses a selection procedure, which is called a *filter*, to generate examples under given distributions. Our filter $\mathbf{FiltEX}_{D_t}$ used at the tth boosting step is standard and essentially the same as the one given in [Fre95]. That is, the filter generates an example $(x, f_*(x))$ by $\mathbf{EX}_{D,f_*}$ and either "rejects" it and throws it away, or "accepts" it and passes it on to **WeakLearn**, where the probability of accepting an example is $w_{t-1}(x)/D(x)$. More precisely, we consider the following procedure.

Procedure $\mathbf{FiltEX}_{D_t}$
  if $W_{t-1} < \epsilon$ **then**
    output "accurate enough" and halt;
  repeat
    use $\mathbf{EX}_{D,f_*}$ to generate an example (x, b);
    $p(x) \leftarrow \min\Big(\prod_{1 \le i \le t-1} \beta_i^{\mathrm{cons}(h_i,x)},\ 1\Big)$;
    % That is, $p(x) = w_{t-1}(x)/D(x)$.
    accept (x, b) with probability $p(x)$;
  until some example (x, b) is accepted;
  return (x, b);
end-procedure.

Then the following properties are immediate.

Lemma 4 *Consider the execution of* $\mathbf{FiltEX}_{D_t}$ *for some* t.
(1) $\mathbf{FiltEX}_{D_t}$ *outputs* x *with probability* $D_t(x)$. *(Recall that* $D_t(x) = w_{t-1}(x)/W_{t-1}$.*)*
(2) The probability that $\mathbf{FiltEX}_{D_t}$ *accepts the first generated example and passes it to* **WeakLearn** *is* $\sum_{x \in X} (w_{t-1}(x)/D(x)) \cdot D(x) = W_{t-1}$.

The above statement (1) guarantees that this filtering procedure $\mathbf{FiltEX}_{D_t}$ gives an example under the desired distribution D_t. On the other hand, it follows from the statement (2) that the expected number of executions of $\mathbf{EX}_{D,f_*}$ until some example is accepted is $1/W_{t-1}$. Thus, the procedure takes longer and longer time when W_{t-1} gets decreased. But also recall that W_{t-1} bounds the error probability of the combined hypothesis; hence, we can stop the filtering procedure if it takes, say, more than $1/\epsilon$ steps to get one example. Therefore, we may assume that the running time of the procedure is $\mathcal{O}(1/\epsilon)$. This is the key of our new weighting scheme.

To completely show that **MadaBoost** can be used in the filtering framework, we also need to consider the following points:- (i) the way to set the confidence values so that the overall algorithm does not fail with probability larger than $1 - \delta$, (ii) the way to determine "$W_{t-1} < \epsilon$" in $\mathbf{FiltEX}_{D_t}$, and (iii) the way to estimate γ_t, the advantage of an obtained weak hypothesis h_t.

To solve (i), notice that the places where the algorithm can fail are three:- in one of the executions of **WeakLearn**, in estimating γ_t, and in determining when to stop (i.e., when W_t becomes less than ϵ). Then in order to obtain an overall confidence of δ, it suffices to guarantee that any of these two procedures fails with probability at most $\delta_t \stackrel{\mathrm{def}}{=} \delta/(3t(t+1))$ at each boosting step t. For (ii), we simply count the number of unsuccessful calls of $\mathbf{EX}_{D,f_*}$ in the execution of $\mathbf{FiltEX}_{D_t}$. We determine $W_{t-1} < \epsilon$ if the number exceeds a certain bound that can be easily derived from an appropriate concentration bound. For instance, it follows from the Hoeffding bound that if $W_{t-1} > \epsilon$, then $\mathbf{FiltEX}_{D_t}$ calls $\mathbf{EX}_{D,f_*}$ more than $\mathcal{O}((1/\epsilon)\ln(1/\delta_t))$ times to yield one example.

Solving (iii) is not trivial. We need the way to estimate the advantage γ_t of an obtained weak hypothesis h_t. Recall that in the proof of Theorem 2, we assumed that γ_t was computed exactly. While γ_t can be directly computed under the subsampling framework, this cannot be done in a straightforward manner in the filtering framework. Notice here, however, that it would be enough if we could obtain an estimate $\widehat{\gamma}_t$ of γ_t such that $|\gamma_t - \widehat{\gamma}_t| \le \gamma_t/2$, because then we could use $\widehat{\gamma}_t/2$ for γ_t. This may slow down the convergence speed a bit, but the proof of Theorem 2 works. Then how can we obtain such an estimator $\widehat{\gamma}_t$ of γ_t? The situation is different from estimating W_{t-1}, and a straightforward application of a convergence bound like the Hoeffding bound does not work. Fortunately, we can make use of more sophisticated estimation methods called "adaptive sampling techniques" that have been proposed in [DGW99, Wat00]. By using one of these techniques, it is possible to obtain a desired $\widehat{\gamma}_t$ from $\mathcal{O}((1/\gamma_t^2)\ln(1/\delta_t))$ examples randomly generated by $\mathbf{EX}_{D,f_*}$.

Now summarizing the above discussion, we have the following theorem.

Theorem 5 *Suppose that for given inputs* $\epsilon > 0$ *and* $\delta < 1$, *the algorithm* **MadaBoost** *used in the filtering framework as described above executes* **WeakLearn** *terminates after the* T*th boosting round, obtaining weak hypotheses with advantages* $\gamma_1 \ge \cdots \ge \gamma_T > 0$.
(1) With probability $1 - \delta$, *we have* $\mathrm{error}_D(X, f_T) \le 1 - \sum_{i=1}^{T} \epsilon\gamma_i^2/8$, *where*

$$\mathrm{error}_D(X, f_T) \stackrel{\mathrm{def}}{=} \Pr_{x:D}\{\, f_T(x) \ne f_*(x)\,\}.$$

(2) With probability $1 - \delta$, *we have* $W_{t-1} \ge \epsilon$ *for any* t, $1 \le t < T$ *(and* $W_T < \epsilon$*) and thus, for each execution of* $\mathbf{FiltEX}_{D_t}$, *the generator* $\mathbf{EX}_{D,f_*}$ *is called*

$\mathcal{O}((1/\epsilon)\ln(1/\delta_t))$ times, where $\delta_t \stackrel{\text{def}}{=} \delta/3t(t+1)$. Similarly, for estimating the advantage of the obtained tth weak hypothesis, $\mathbf{EX}_{D,f_*}$ is called $\mathcal{O}((1/\epsilon\gamma_t^2)((\ln(1/\delta_t))^2)$ times.

We can now explain clearly why **AdaBoost** cannot be used for the filtering framework. As it happens with **MadaBoost**, the boosting property of **AdaBoost**, i.e., Theorem 1, still holds in the filtering framework. Furthermore, W_{t-1} also bounds the generalization error of the tth combined hypothesis (This property only holds under the weighting scheme as defined in Section 2). On the other hand, we cannot use the above filtering procedure as it is because the weights are not bounded between 0 and 1. There are two ways to get around this, either we normalize the weights of **AdaBoost** or we use the original definition in [FS97] where the weights are bounded between 0 and 1. Both ways will lead to the same conclusion; that is, the probability that the filter outputs one example could be exponentially smaller than the current generalization error. For example, in the case that the advantage of all obtained weak hypotheses is γ, we can show that the expected running time of the filter to output one example becomes exponential in $1/\gamma$.

5 Noise Tolerance

We show here, from the theoretical point of view, that **MadaBoost** is in fact resistant to certain kinds of noise by showing that it belongs to the statistical query model of learning introduced by Kearns [Kea93].

Before showing the result, let us recall the definitions of the statistical query model. In the Statistical Query (SQ) model a learner does not have access to examples anymore. The example oracle $EX(f_*, D)$ is replaced by a statistics oracle $STAT(f_*, D)$ that works as follows. A statistical query is of the form $[\chi, \tau]$, where τ is a tolerance parameter, and χ is a mapping from labeled examples to $\{0,1\}$ (i.e. $\chi : X \times \{0,1\} \to \{0,1\}$) corresponding to an indicator function for those examples about which statistic are to be collected. A call to $STAT(f_*, D)[\chi, \tau]$ returns an estimate $\widehat{P}_\chi$ of $P_\chi = \Pr_{x:D}\{\,\chi(x, f_*(x))\,\}$ that satisfies $|\widehat{P}_\chi - P_\chi| \le \tau$. By using queries of this type, we would like to design a learning algorithm that is robust to random classification noise.

Now we show that **MadaBoost** belongs to the SQ model. That is, the following theorem. (The same statement holds for **MB:1/2**.)

Theorem 6 *Algorithm* **MadaBoost** *is an statistical query boosting algorithm.*

We first give a proof outline. As in [AD93], we assume that **WeakLearn** is a SQ learning algorithm; that is, it uses the statistical oracle instead of the example oracle. Notice here that **WeakLearn** works under the modified distributions. Hence at each step t it does not use $STAT(f_*, D)$ but $STAT(f_*, D_t)$. Therefore for the proof it suffices to show that we can simulate a query to $STAT(f_*, D_t)$ by using queries to $STAT(f_*, D)$ that is the oracle available to the boosting algorithm. More specifically, we fix a query $STAT(f_*, D_t)[\chi, \tau]$ and assume that we are in the tth boosting step and that **WeakLearn** is making the query $STAT(f_*, D_t)[\chi, \tau]$ in order to construct hypothesis h_{t+1}. Then we show the way to simulate $STAT(f_*, D_t)[\chi, \tau]$ with queries to $STAT(f_*, D)$. Here we also assume that $W_t \ge \epsilon$ for a given parameter ϵ the boosting algorithm, because otherwise, the boosting process would have been terminated.

Now we explain our simulation of $STAT(f_*, D_t)[\chi, \tau]$ in detail. For our analysis, we need to divide a subspace $X' \stackrel{\text{def}}{=} \{\, x \in X \mid f_t(x) = f_*(x)\,\}$ in slices depending on the value of $w_t(x)$. For this, we define n as the smallest integer satisfying the following inequality, and Y as follows.

$$\log\left(\frac{96(n+1)^3}{\tau\epsilon}\right) \le n, \text{ and } Y \stackrel{\text{def}}{=} 1 - \frac{\tau\epsilon}{192(n+1)^2}.$$

Let l be the smallest integer such that $Y^l \le \prod_{i=1}^{t} \beta_i$ holds. Then define the following set for each $k \ge 0$. (Note that $\{E_k\}_{k\ge 0}$ is the division of X'.)

$$E_k \stackrel{\text{def}}{=} \{\, x \mid x \in X \text{ and } Y^{k+1} \le \prod_{i=1}^{t} \beta_i^{\text{cons}(h_i,x)} < Y^k \,\}.$$

Armed with this definition, we can now rewrite the query $STAT(f_*, D_t)[\chi, \tau]$ as follows.

$$\begin{aligned}
STAT(f_*, D_t)[\chi, \tau] &= \Pr_{x:D_t}\{\,\chi(x, f_*(x))\,\} \\
&= \sum_{x\in X} \chi(x, f_*(x)) D_t(x) \\
&= \sum_{\substack{x\in X \\ f_t(x)=f_*(x)}} \chi(x, f_*(x)) D_t(x) \\
&\quad + \sum_{\substack{x\in X \\ f_t(x)\neq f_*(x)}} \chi(x, f_*(x)) D_t(x) \\
&= \sum_{0\le k\le l-1} \sum_{x\in E_k} \chi(x, f_*(x)) D_t(x) \\
&\quad + \sum_{\substack{x\in X \\ f_t(x)\neq f_*(x)}} \chi(x, f_*(x)) \frac{D(x)}{W_t} \\
&= \sum_{0\le k\le l} \Pr_{x:D_t}\{\,\chi(x, f_*(x)) \wedge x \in E_k\,\} \\
&\quad + \frac{\Pr_{x:D}\{\,\chi(x, f_*(x)) \wedge f_t(x) \neq f_*(x)\,\}}{W_t}
\end{aligned}$$

Now we want to approximate these two additive terms of the above formula. We show how each term can be approximated efficiently up to an additive tolerance of $\tau/2$; then the overall probability is approximated up to an additive tolerance of τ as desired.

The second term in the above is already given in terms of probability with respect to D; hence, we can use $STAT(f_*, D)$ to approximate it appropriately as shown by the following lemma. (The proof is easy and omitted here.)

Lemma 7 *Let $p = \Pr_{x:D}\{\,\chi(x, f_*(x)) \wedge f_t(x) \neq f_*(x)\,\}$ and let $\widehat{p}$ and $\widehat{W_t}$ be two estimators such that $|\,\widehat{p} - p\,| \le \tau W_t/6$ and $|\,\widehat{W_t} - W_t\,| \le \tau W_t/6$, then an estimator $\widehat{p}/\widehat{W_t}$ satisfies*

$$\left|\, \frac{\widehat{p}}{\widehat{W_t}} - \frac{p}{W_t} \,\right| \le \frac{\tau}{2}.$$

For the first term, let us define the following probabilities in order to simplify the notation.

$$p_k^{(1)} \stackrel{\text{def}}{=} \Pr_{x:D_t}\{\,\chi(x, f_*(x)) \wedge x \in E_k\,\}, \text{ and}$$
$$p_k^{(2)} \stackrel{\text{def}}{=} \Pr_{x:D}\{\,\chi(x, f_*(x)) \wedge x \in E_k\,\}.$$

Then for any $k \geq 0$, the following inequalities hold by definition of E_k.

$$\frac{Y^{k+1}p_k^{(2)}}{W_t} \;\leq\; p_k^{(1)} \;\leq\; \frac{Y^k p_k^{(2)}}{W_t}. \tag{1}$$

Thus, our goal is to obtain, for every k, an estimator $\widehat{p}_k^{(1)}$ that approximates $p_k^{(1)}$ up to an appropriate additive tolerance. We estimate $p_k^{(1)}$ by using the estimators $\widehat{p}_k^{(2)}$ and $\widehat{W}_t$ of $p_k^{(2)}$ and W_t. The following lemma states the way.

Lemma 8 *For any $k \geq 0$, let $\widehat{p}_k^{(2)}$ and $\widehat{W}_t$ two estimators satisfying*

$$|\,\widehat{p}_k^{(2)} - p_k^{(2)}\,| \;\leq\; \frac{\tau W_t}{24Y^{k+1}(n+1)}, \text{ and}$$
$$|\,\widehat{W}_t - W_t\,| \;\leq\; \frac{\tau W_t}{24(n+1)}.$$

Then by defining $\widehat{p}_k^{(1)} \stackrel{\text{def}}{=} Y^{k+1}\widehat{p}_k^{(2)}/\widehat{W}_t$, we have

$$|\,\widehat{p}_k^{(1)} - p_k^{(1)}\,| \;\leq\; \frac{\tau}{4(n+1)}.$$

Proof. Since $W_t \geq \epsilon$, we have $1 - Y = \tau\epsilon/(192(n+1)^2) \leq \tau W_t/8(n+1)$. Moreover, since $p_k^{(2)}$ and Y^k are both less than or equal to 1, the following inequality holds.

$$Y^k(1-Y)p_k^{(2)} \;\leq\; \frac{\tau W_t}{8(n+1)},$$

which implies

$$\frac{Y^k p_k^{(2)}}{W_t} - \frac{Y^{k+1}p_k^{(2)}}{W_t} \;\leq\; \frac{\tau}{8(n+1)}.$$

This inequality together with inequality (1) implies

$$p_k^{(1)} - \frac{\tau}{4(n+1)} \;\leq\; \frac{Y^{k+1}p_k^{(2)}}{W_t} - \frac{\tau}{8(n+1)}, \text{ and}$$
$$\frac{Y^{k+1}p_k^{(2)}}{W_t} + \frac{\tau}{8(n+1)} \;\leq\; p_k^{(1)} + \frac{\tau}{4(n+1)}.$$

Thus, if we show that estimator $\widehat{p}_k^{(1)}$ as defined in the statement of the lemma approximates $Y^{k+1}p_k^{(2)}/W_t$ up to an additive tolerance smaller than $\tau/(8(n+1))$, then, by the inequality above, we can conclude that $\widehat{p}_k^{(1)}$ also approximates $p_k^{(1)}$ up to an additive tolerance smaller than $\tau/(4(n+1))$ as claimed in the statement of the lemma.

First note the following two inequalities, which are easy to show.

$$\frac{Y^{k+1}p_k^{(2)} + \frac{\tau W_t}{24(n+1)}}{W_t - \frac{\tau W_t}{24(n+1)}} \;\leq\; \frac{Y^{k+1}p_k^{(2)}}{W_t} + \frac{\tau}{8(n+1)}, \text{ and}$$
$$\frac{Y^{k+1}p_k^{(2)}}{W_t} - \frac{\tau}{8(n+1)} \;\leq\; \frac{Y^{k+1}p_k^{(2)} - \frac{\tau W_t}{24(n+1)}}{W_t + \frac{\tau W_t}{24(n+1)}}.$$

Next use the assumption of the lemma on $\widehat{p}_k^{(2)}$ and $\widehat{W}_t$; then we have

$$\frac{Y^{k+1}p_k^{(2)} - \frac{\tau W_t}{24(n+1)}}{W_t + \frac{\tau W_t}{24(n+1)}} \;\leq\; \frac{Y^{k+1}\widehat{p}_k^{(2)}}{\widehat{W}_t} \;\leq\; \frac{Y^{k+1}p_k^{(2)} + \frac{\tau W_t}{24(n+1)}}{W_t - \frac{\tau W_t}{24(n+1)}}.$$

This implies that $|\widehat{p}_k^{(1)} - Y^{k+1}p_k^{(2)}/W_t| \leq \tau/8(n+1)$, proving the lemma as already argued. □

Notice that as k becomes large the probability $p_k^{(1)}$ gets smaller. In fact, we do not need to approximate all of them; only the first $n+1$ would be enough. The rest can be discarded as shown in the following lemma.

Lemma 9 $\sum_{n+1\leq k\leq l-1} p_k^{(1)} \;\leq\; \tau/4.$

Proof. Since $Y < 1$ and $\epsilon \leq W_t$, it follows from our choice of n that $\log(96(k+1)^3/(\tau W_t)) \leq k\log(1/Y)$ for any $k > n$. This implies

$$\frac{Y^k}{W_t} \;\leq\; \frac{\tau}{96(k+1)^3} \;\leq\; \frac{\tau}{4k^2}.$$

Moreover, the following chain of inequalities also hold.

$$\sum_{n+1\leq k\leq l-1} p_k^{(1)} \;\leq\; \sum_{n+1\leq k\leq l-1} \frac{Y^k p_k^{(2)}}{W_t} \;\leq\; \sum_{n+1\leq k\leq l-1} \frac{Y^k}{W_t} \;\leq\; \sum_{n+1\leq k\leq l-1} \frac{\tau}{4k^2}.$$

Then the lemma follows by noticing that $\sum_{n+1\leq k\leq l-1} \tau/4k^2$ is smaller than $\tau/4$. □

Thus, for $1 \leq k \leq n$, we use the estimator $\widehat{p}_k^{(1)}$ as defined in Lemma 8, and for any $k > n$, we just use $\widehat{p}_k^{(1)} = 0$ as our estimator. In this way, we can bound the error as follows.

$$\left|\sum_{0\leq k\leq l-1} p_k^{(1)} - \sum_{0\leq k\leq l-1} \widehat{p}_k^{(1)}\right| \;\leq\; \sum_{0\leq k\leq n} \frac{\tau}{4(n+1)} + \sum_{n+1\leq k\leq l-1} \frac{\tau}{4k^2} \;\leq\; \frac{\tau}{2}.$$

Estimators $\widehat{p}$ and $\widehat{p}_k^{(2)}$ can be obtained from queries to $STAT(f_*, D)$ since they are estimators of probabilities with respect to D, and the conditions $f_t(x) \neq f_*(x)$ and $x \in E_k$ can be tested in polynomial time. On the other hand, $\widehat{W}_t$ cannot be directly obtained from $STAT$. Before describing

how to obtain $\widehat{W}_t$, let us discuss about the tolerances of these estimators.

We need to verify that the tolerances required for all the approximations involved in the proof are not too small. In other words, we need to show that the inverse of all the tolerances are polynomial in $1/\epsilon$ and $1/\tau$. First recall that all the tolerances are required to depend on W_t. This value is not known but we can assume that it is larger than ϵ throughout all the boosting process; hence, we can substitute W_t by ϵ in all the tolerances. The tolerance required for $\widehat{W}_t$ in Lemma 8 (i.e., $\tau\epsilon/24(n+1)$) is the smallest among all the tolerances required; thus, it is enough to show that $24(n+1)/\tau\epsilon$ is bounded by a polynomial in $1/\epsilon$ and $1/\tau$. But this follows from the fact that n is bounded by a polynomial; more specifically, it is easy to show that n is bounded by $\mathcal{O}((1/\epsilon\tau)\ln(1/\epsilon\tau))$.

It remains to see how can we approximate W_t from $STAT(f_*, D)$. The smallest tolerance required for W_t through the proof is $\tau\epsilon/24(n+1)$; hence, we show how to approximate W_t up to this tolerance. First we rewrite W_t as follows.

$$\begin{aligned} W_t &= \sum_{x\in X} w_t(x) \\ &= \Pr_{x:D}\{\, f_t(x) \neq f_*(x)\,\} + \sum_{0\le k\le l-1}\ \sum_{x\in E_k} w_t(x). \end{aligned}$$

Moreover, by definition of E_k, the following inequality holds for any $k \ge 0$.

$$Y^{k+1} q_k^{(2)} \ \le\ q_k^{(1)} \ \le\ Y^k q_k^{(2)},$$

where $q_k^{(2)} = \Pr_{x:D}\{\, x \in E_k \,\}$ and $q_k^{(1)} = \sum_{x\in E_k} w_t(x)$.

The following lemma shows how to approximate W_t. (The proof, which is similar to the previous two proofs, is omitted.)

Lemma 10 *Let $q = \Pr_{x:D}\{\, f_t(x) \neq f_*(x)\,\}$ and for any $k \ge 0$, let $\widehat{q}_k^{(2)}$ and $\widehat{q}$ be two estimators such that*

$$|\,\widehat{q}_k^{(2)} - q_k^{(2)}\,| \ \le\ \frac{\tau\epsilon}{Y^{k+1}192(n+1)^2} \ \text{ and } \ |\,\widehat{q} - q\,| \ \le\ \frac{\tau\epsilon}{48n}.$$

Then by defining $\widehat{W}_t \stackrel{\text{def}}{=} \widehat{q} + \sum_{0\le k\le n} \widehat{q}_k^{(2)}$, we have

$$|\,W_t - \widehat{W}_t\,| \ \le\ \frac{\tau\epsilon}{24(n+1)}.$$

Summarizing, we have shown how to estimate $\Pr_{x:D_t}\{\, \chi(x, f_*(x))\,\}$ up to tolerance τ by combining appropriately the answers of the following statistical queries.

$STAT(f_*, D)[\,\chi(x, f_*(x)) \wedge f_t(x) \neq f_*(x)\,,\ \tau\epsilon/6\,]$
$STAT(f_*, D)[\,\chi(x, f_*(x)) \wedge x \in E_k\,,\ \tau\epsilon/(24Y^{k+1}(n+1))\,]$
(for each k, $0 \le k \le n$)
$STAT(f_*, D)[\,f_t(x) \neq f_*(x)\,,\ \tau\epsilon/(48n)\,]$, and
$STAT(f_*, D)[\,x \in E_k\,,\ \tau\epsilon/(192Y^{k+1}(n+1)^2)\,]$
(for each k, $0 \le k \le n$).

Therefore, we have proven Theorem 6.

References

[AL88] D. Angluin and P. Laird, Learning from noisy examples, *Machine Learning*, 2(4):343–370, 1988.

[AD93] J.A. Aslam and S.E. Decatur, General bounds on statistical query learning and PAC learning with noise via hypothesis boosting, in *Proc. of the 34th Annual Sympos. on Foundations of Comp. Sci.*, 282–291, 1993.

[Die98] T.G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting and randomization, *Machine Learning*, 32:1–22, 1998.

[DKM96] T. Dietterich, M. Kearns, and Y. Mansour, Applying the weak learning framework to understand and improve C4.5, in *Proc. 13th International Conference on Machine Learning*, 96–104, 1996.

[DGW99] C. Domingo, R. Gavaldà, and O. Watanabe, Adaptive sampling methods for scaling up knowledge discovery algorithms, in *Proc. of the Second International Conference on Discovery Science*, DS'99, Lecture Notes in Artificial Intelligence 1721, 172–183, 1999.

[DW99] C. Domingo and O. Watanabe, Experimental evaluation of a modified AdaBoost for the filtering framework, Technical report C-139, Dept. of Mathematics and Computer Science, Tokyo Institute of Technology, available from **www.is.titech.ac.jp/research/research-report/C/index.html**, 1999.

[DW00] C. Domingo and O. Watanabe, Scaling up a boosting-based learner via adaptive sampling, in *Proc. of Knowledge Discovery and Data Mining*, PAKDD'00, Lecture Notes in Artificial Intelligence 1805, 317–328, 2000

[Fre95] Y. Freund, Boosting a weak learning algorithm by majority, *Information and Computation*, 121(2):256–285, 1995.

[Fre99] Y. Freund, An adaptive version of the boost by majority algorithm, in *Proc. of the Twelfth Annual Conference on Computational Learning Theory*, 1999.

[FS97] Y. Freund and R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[FS96] Y. Freund and R.E. Schapire, Experiments with a new boosting algorithm, in *Proc. of the Thirteenh International Conference on Machine Learning*, 148–156, 1996.

[FHT98] J. Friedman, T. Hastie, and R. Tibshirani, Additive logistic regression: a statistical view of boosting, Technical report, 1998.

[Kea93] M. Kearns, Efficient noise-tolerant learning from statistical queries, In *Proc. of the Twenty-Fifth Annual ACM Sympos. on Theory of Comput.*, 392–401, 1993.

[Sch90] R.E. Schapire, The strength of weak learnability, *Machine Learning*, 5(2):197–227, 1990.

[Wat99] O. Watanabe, From computational learning theory to discovery science, in *Proc. 26th International Colloquium on Automata, Languages and Programming*, ICALP'99, Lecture Notes in Computer Science 1644, 134–148, 1999.

[Wat00] O. Watanabe, Simple sampling techniques for discovery science, *IEICE Trans. on Information and Systems*, E83-D(1): 19–26, 2000.