PAC Analogues of Perceptron and Winnow via Boosting the Margin

Rocco A. Servedio[*]
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
rocco@deas.harvard.edu

Abstract

We describe a novel family of PAC model algorithms for learning linear threshold functions. The new algorithms work by boosting a simple weak learner and exhibit complexity bounds remarkably similar to those of known online algorithms such as Perceptron and Winnow, thus suggesting that these well-studied online algorithms in some sense correspond to instances of boosting. We show that the new algorithms can be viewed as natural PAC analogues of the online p-norm algorithms which have recently been studied by Grove, Littlestone, and Schuurmans [16] and Gentile and Littlestone [15]. As special cases of the algorithm, by taking $p = 2$ and $p = \infty$ we obtain natural boosting-based PAC analogues of Perceptron and Winnow respectively. The $p = \infty$ case of our algorithm can also be viewed as a generalization (with an improved sample complexity bound) of Jackson and Craven's PAC-model boosting-based algorithm for learning "sparse perceptrons" [20]. The analysis of the generalization error of the new algorithms relies on techniques from the theory of large margin classification.

1 INTRODUCTION

One of the most fundamental problems in computational learning theory is that of learning an unknown linear threshold function from labeled examples. Many different learning algorithms for this problem have been considered over the past several decades. In particular, in recent years many researchers have studied simple online additive and multiplicative update algorithms, namely the Perceptron and Winnow algorithms and variants thereof [3, 5, 8, 14, 15, 16, 25, 26, 27, 28, 33, 36].

This paper takes a different approach. We describe a natural parameterized family of boosting-based PAC algorithms for learning linear threshold functions. The weak hypotheses used are linear functionals and the strong classifier obtained is a linear threshold function. Although these new algorithms are conceptually and algorithmically very different from Perceptron and Winnow, we establish performance bounds for the new algorithms which are remarkably similar to those of Perceptron and Winnow; we thus refer to the new algorithms as *PAC analogues* of Perceptron and Winnow. We hope that the analysis of these new algorithms will yield fresh insights into the relationship between boosting and online algorithms.

We give a unified analysis of our Perceptron and Winnow analogues which includes many other algorithms as well. Grove, Littlestone and Schuurmans [16] have shown that Perceptron and (a version of) Winnow can be viewed as the $p = 2$ and $p \to \infty$ cases of a general online p-norm linear threshold learning algorithm, where $p \geq 2$ is any real number. We present PAC-model boosting-based analogues of these online p-norm algorithms for any value $2 \leq p \leq \infty$. The PAC-model Perceptron and Winnow analogues mentioned above are respectively the $p = 2$ and $p = \infty$ cases of this general algorithm.

The $p = \infty$ case of our algorithm can also be viewed as a generalization of Jackson and Craven's PAC-model algorithm for learning "sparse perceptrons" [20]. Their algorithm boosts using weak hypotheses which are single Boolean literals; this is similar to what the $p = \infty$ case of our algorithm does. Our analysis of the $p = \infty$ case generalizes their algorithm to deal with real-valued rather than Boolean input variables and yields a substantially stronger sample complexity bound than was established in [20].

Section 2 of this paper contains preliminary material, including an overview of the online p-norm algorithms from [15, 16]. In Section 3 we present a simple PAC-model p-norm algorithm and prove that it is a weak learning algorithm for all $2 \leq p < \infty$. In Section 4 we apply techniques from the theory of large margin classification to show how our weak learning algorithm can be boosted to a strong learning algorithm with small sample complexity. Finally, in Section 5 we compare our PAC algorithms with the analogous online algorithms, extend our algorithm to the case $p = \infty$, and discuss the relationship between the $p = \infty$ case of our algorithm and the Jackson–Craven algorithm for learning sparse perceptrons.

1.1 RELATED WORK

Several authors have studied linear threshold learning algorithms which work by combining weak predictors. Freund and Schapire [14] describe an algorithm which combines intermediate Perceptron algorithm hypotheses using a weighted

[*] Supported in part by an NSF Graduate Fellowship, by NSF grant CCR-95-04436 and by ONR grant N00014-96-1-0550.

majority vote (so the final classifier is a depth-2 threshold circuit) and prove bounds on the generalization error of the resulting classifier. Their algorithm does not use boosting to combine the Perceptron hypotheses but rather weights them according to their survival time. Ji and Ma [21] propose a random-search-and-test approach to find weak classifier linear threshold functions and combine them by a simple majority vote (thus also obtaining a depth-2 threshold circuit). Our approach is closest to that of Jackson and Craven [20] who use boosting to combine single literals into a strong hypothesis linear threshold function. As described in Section 5, the $p = \infty$ case of our algorithm strengthens and generalizes their results. More generally, we also note that Freund and Schapire [12] and Schapire [32] have exhibited a close relationship between boosting and online learning.

2 PRELIMINARIES

We start with some geometric definitions. For a point $\tilde{x} = (x_1, \ldots, x_n) \in \Re^n$ and $p \geq 1$ we write $\|\tilde{x}\|_p$ to denote the p-norm of $\tilde{x}$, namely

$$\|\tilde{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}.$$

The ∞-norm of $\tilde{x}$ is $\|\tilde{x}\|_\infty = \max_{i=1,\ldots,n} |x_i|$. For $p, q \geq 1$ the q-norm is *dual* to the p-norm if $\frac{1}{p} + \frac{1}{q} = 1$; hence the 1-norm and the ∞-norm are dual to each other and the 2-norm is dual to itself. In this paper p and q always denote dual norms. The following facts are well known (e.g. [37] pp. 203-204):

Hölder Inequality: $|\tilde{u} \cdot \tilde{v}| \leq \|\tilde{u}\|_p \|\tilde{v}\|_q$ for all $\tilde{u}, \tilde{v} \in \Re^n$ and $1 \leq p \leq \infty$.

Minkowski Inequality: $\|\tilde{u} + \tilde{v}\|_p \leq \|\tilde{u}\|_p + \|\tilde{v}\|_p$ for all $\tilde{u}, \tilde{v} \in \Re^n$ and $1 \leq p \leq \infty$.

Throughout this paper the *example space* X is a subset of $\Re^n$. A *linear threshold function* over X is a function f such that $f(\tilde{x}) = \text{sign}(\tilde{u} \cdot \tilde{x})$ for some $\tilde{u} \in \Re^n$ (recall that the function $\text{sign}(z)$ takes value 1 if $z \geq 0$ and takes value -1 if $z < 0$). We note that the standard definition of a linear threshold function allows a nonzero threshold, i.e. $f(\tilde{x}) = \text{sign}(\tilde{u} \cdot \tilde{x} - \theta)$ where θ can be any real number. However, any linear threshold function of this more general form over n variables is equivalent to a linear threshold function with threshold 0 over $n + 1$ variables, so our definition incurs no real loss of generality.

We write $\|X\|_p$ to denote $\sup_{\tilde{x} \in X} \|\tilde{x}\|_p$. We use the symbol $\delta_{\tilde{u},X}$ to denote the quantity

$$\delta_{\tilde{u},X} \stackrel{\text{def}}{=} \inf_{\tilde{x} \in X} (\tilde{u} \cdot \tilde{x})(\text{sign}(\tilde{u} \cdot \tilde{x})),$$

which is a measure of the separation between examples in X and the hyperplane whose normal vector is $\tilde{u}$. We assume throughout the paper that $\|X\|_p < \infty$, i.e. the set X is bounded, and that $\delta_{\tilde{u},X} > 0$, i.e. there is some nonzero lower bound on the separation between the hyperplane defined by $\tilde{u}$ and the examples in X.

2.1 PAC LEARNING

For $\tilde{u} \in \Re^n$ let $EX(\tilde{u}, \mathcal{D})$ denote an *example oracle* which, when queried, provides a labeled example $\langle \tilde{x}, \text{sign}(\tilde{u} \cdot \tilde{x}) \rangle$ where $\tilde{x}$ is drawn according to the distribution $\mathcal{D}$ over X. We say that an algorithm A is a *strong learning algorithm for* $\tilde{u}$ *on* X if it satisfies the following condition: there is a function $m(\epsilon, \delta, \tilde{u}, X)$ such that for any distribution $\mathcal{D}$ over X, for all $0 < \epsilon, \delta < 1$, algorithm A makes at most $m(\epsilon, \delta, \tilde{u}, X)$ calls to $EX(\tilde{u}, \mathcal{D})$, and with probability at least $1 - \delta$ algorithm A outputs a hypothesis $h : X \to \{-1, 1\}$ such that $\Pr_{x \in \mathcal{D}}[h(\tilde{x}) \neq \text{sign}(\tilde{u} \cdot \tilde{x})] \leq \epsilon$. We say that such a hypothesis h is an ϵ*-accurate hypothesis for* $\tilde{u}$ *under* $\mathcal{D}$ and that the function $m(\epsilon, \delta, \tilde{u}, X)$ is the *sample complexity* of algorithm A.

As our main result we describe a strong learning algorithm and carefully analyze its sample complexity. To do this we must consider algorithms which do not satisfy the strong learning property but are still capable of generating hypotheses that have some slight advantage over random guessing (such so-called weak learning algorithms were first considered by Kearns and Valiant in [24]). Let

$$S = \langle \tilde{x}^1, \text{sign}(\tilde{u} \cdot \tilde{x}^1) \rangle, \ldots, \langle \tilde{x}^m, \text{sign}(\tilde{u} \cdot \tilde{x}^m) \rangle$$

be a finite sequence of labeled examples from X and let $\mathcal{D}$ be a distribution over S. For $0 < \gamma < 1/2$, we say that $h : X \to [-1, 1]$ is a $(1/2 - \gamma)$*-approximator for* $\tilde{u}$ *under* $\mathcal{D}$ if

$$\frac{1}{2} \sum_{i=1}^{m} \mathcal{D}(\tilde{x}^i) \cdot |h(\tilde{x}^i) - \text{sign}(\tilde{u} \cdot \tilde{x}^i)| \leq \frac{1}{2} - \gamma. \qquad (1)$$

We say that an algorithm A is a $(1/2 - \gamma)$*-weak learning algorithm for* $\tilde{u}$ *under* $\mathcal{D}$ if the following condition holds: for any finite set S as described above and any distribution $\mathcal{D}$ on S, if A is given $\mathcal{D}$ and S as input then A outputs a hypothesis $h : X \to [-1, 1]$ which is a $(1/2 - \gamma)$-approximator for $\tilde{u}$ under $\mathcal{D}$. Thus for our purposes a weak learning algorithm is one which can always find a hypothesis that outperforms random guessing on a fixed sample.

2.2 ONLINE LEARNING AND p-NORM ALGORITHMS

In the *online* model, learning takes place over a sequence of trials. Throughout the learning process the learner maintains a hypothesis h which maps X to $\{-1, 1\}$. Each trial proceeds as follows: upon receiving an example $x \in X$ the learning algorithm outputs its prediction $\hat{y} = h(x)$ of the associated label y. The learning algorithm is then given the true label $y \in \{-1, 1\}$ and the algorithm can update its hypothesis h based on this new information before the next trial begins. The performance of an online learning algorithm on an example sequence is measured by the number of prediction mistakes which the algorithm makes.

Grove, Littlestone and Schuurmans [16] and Gentile and Littlestone [15] have studied a family of online algorithms for learning linear threshold functions (see Figure 1). We refer to this algorithm, which is parameterized by a real value $p \geq 2$, as the *online p-norm algorithm.* Like the well-known Perceptron algorithm, the online p-norm algorithm updates its hypothesis by making an additive change to a weight vector $\tilde{z}$. However, as shown in steps 4-5 of Figure 1, the p-norm

```
Input parameter: real number $p \geq 2$, initial weight vector $\tilde{z}^0 = (z_1^0, \ldots, z_n^0) \in \Re^n$, positive value $a > 0$
set $t = 0$
while examples are available do
    get unlabeled example $\tilde{x}^t$
    for all $i = 1, \ldots, n$ set $w_i^t = \text{sign}(z_i^t)|z_i^t|^{p-1}$
    predict $\hat{y}_t = \text{sign}(\tilde{w}^t \cdot \tilde{x}^t)$
    get label $y_t \in \{-1, +1\}$
    for all $i = 1, \ldots, n$ set $z_i^{t+1} = z_i^t + a(y_t - \hat{y}_t)x_i^t$
    set $t = t + 1$
enddo
```

Figure 1: The online p-norm algorithm.

algorithm does not use the $\tilde{z}$ vector directly for prediction but rather predicts using a vector $\tilde{w}$ which is a transformed version of the $\tilde{z}$ vector, namely $w_i = \text{sign}(z_i)|z_i|^{p-1}$ for all $i = 1, \ldots, n$. Note that when $p = 2$ we have $\tilde{z} = \tilde{w}$ and hence the online 2-norm algorithm is the Perceptron algorithm. In [16] it is shown that as $p \to \infty$ the online p-norm algorithm approaches a version of the Winnow algorithm. More precisely, the following theorem from [16] gives mistake bounds for the online p-norm algorithms:

Theorem 1 *Let $S = \langle \tilde{x}^1, y_1 \rangle, \ldots, \langle \tilde{x}^m, y_m \rangle$ be a sequence of labeled examples where $\tilde{x} \in X$ and $y = \text{sign}(\tilde{u} \cdot \tilde{x})$ for every example $\langle \tilde{x}, y \rangle \in S$.*

(a) For any $2 \leq p < \infty$ and any $a > 0$, if the online p-norm algorithm is invoked with input parameters $(p, \tilde{z}^0 = (0, \ldots, 0), a)$, then the mistake bound on the example sequence S is at most

$$\frac{(p-1)\|\tilde{u}\|_q^2 \|X\|_p^2}{\delta_{\tilde{u},X}^2}.$$

(b) For any $2 \leq p < \infty$, if $\tilde{z}^0$ satisfies $\tilde{u} \cdot \tilde{z}^0 > 0$ and $a = \frac{\delta_{\tilde{u},X}\|\tilde{z}^0\|_p^2}{(p-1)\tilde{u}\cdot\tilde{z}^0\|X\|_p^2}$, then the mistake bound on S is at most

$$\frac{(p-1)\|\tilde{u}\|_q^2 \|X\|_p^2}{\delta_{\tilde{u},X}^2}\left(1 - \left(\frac{\tilde{u} \cdot \tilde{z}^0}{\|\tilde{u}\|_q \|\tilde{z}^0\|_p}\right)^2\right).$$

(c) Let $\tilde{z}^0 = (1, \ldots, 1)$ and suppose that $u_i > 0$ for $i = 1, \ldots, n$. If $p \to \infty$ and a is as described in part (b), then the mistake bound given in (b) converges to

$$\frac{2\|\tilde{u}\|_1^2 \|X\|_\infty^2}{\delta_{\tilde{u},X}^2}\left(\log n + \sum_{i=1}^n \frac{u_i}{\|\tilde{u}\|_1} \log \frac{u_i}{\|\tilde{u}\|_1}\right).$$

2.3 FROM ONLINE TO PAC LEARNING

Various generic procedures have been proposed [1, 18, 22] for automatically converting on-line learning algorithms into PAC-model algorithms. In these procedures the sample complexity of the resulting PAC algorithm depends on the mistake bound of the original on-line learning algorithm. The strongest general result of this type (in terms of minimizing the sample complexity of the resulting PAC algorithm) is the "longest-survivor" conversion due to Kearns, Li, Pitt and Valiant[1] [22]:

Theorem 2 *Let A be an on-line learning algorithm which is guaranteed to make at most M mistakes. Then there is a PAC-model learning algorithm A' which uses*

$$O\left(\frac{M}{\epsilon}\left(\log \frac{1}{\delta} + \log M\right)\right)$$

examples and outputs an ϵ-accurate hypothesis with probability $1 - \delta$.

Theorems 1 and 2 yield sample complexity bounds on a generic PAC-model conversion of the online p-norm algorithm. We now describe a completely different PAC-model algorithm which has remarkably similar sample complexity bounds.

3 A PAC-MODEL p-NORM WEAK LEARNING ALGORITHM

The p-norm weak learning algorithm is motivated by the following simple idea: Suppose that $S = \langle \tilde{x}^1, y_1 \rangle, \ldots, \langle \tilde{x}^m, y_m \rangle$ is a collection of labeled examples where $y_i = \text{sign}(\tilde{u} \cdot \tilde{x}^i)$ for each $i = 1, \ldots, m$. Now imagine replacing each negative example $\langle \tilde{x}^i, -1 \rangle$ in S by the equivalent positive example $\langle -\tilde{x}^i, 1 \rangle$ to obtain a new collection S' of examples. Let $\tilde{z} \in \Re^n$ be the average location of an example in S', i.e. $\tilde{z}$ is the "center of mass" of S'. Since every example in S' must lie on the same side of the hyperplane $\tilde{u} \cdot \tilde{x} = 0$ as the vector $\tilde{u}$, it is clear that $\tilde{z}$ must also lie on this side of the hyperplane. One might even hope that $\tilde{z}$, or some related vector, points in approximately the same direction as the vector $\tilde{u}$.

Our p-norm weak learning algorithm, which we call `WLA`, is presented in Figure 2. As in the online p-norm algorithm, `WLA` transforms the vector $\tilde{z}$ to a vector $\tilde{w}$ using the mapping $w_i = \text{sign}(z_i)|z_i|^{p-1}$. We now show that this simple algorithm is in fact a weak learner:

Theorem 3 `WLA` *is a $(1/2 - \gamma)$-weak learning algorithm for $\tilde{u}$ under $\mathcal{D}$ for $\gamma = \frac{\delta_{\tilde{u},X}}{2\|X\|_p\|\tilde{u}\|_q}$.*

[1] Littlestone [27] gives a conversion procedure which yields a PAC sample complexity bound of $O(\epsilon^{-1}(\log \delta^{-1} + M))$. Although this improves on the result of [22] by a $\log M$ factor, Littlestone's procedure requires the example space X to be finite, which is a stronger assumption than we make in this paper.

```
Input parameters: real number $p \geq 2$, sequence $S = \langle \tilde{x}^1, y_1 \rangle, \ldots, \langle \tilde{x}^m, y_m \rangle$ of labeled examples, distribution $\mathcal{D}$ over $S$
    1.  set $\tilde{z} = \sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) y_j \tilde{x}^j$
    1.  for all $i = 1, \ldots, n$ set $w_i = \text{sign}(z_i)|z_i|^{p-1}$
    2.  return hypothesis $h(\tilde{x}) \equiv \frac{\tilde{w} \cdot \tilde{x}}{\|\tilde{w}\|_q \|X\|_p}$
```

Figure 2: The p-norm weak learning algorithm WLA.

Proof: Let $S = \langle \tilde{x}^1, y_1 \rangle, \ldots, \langle \tilde{x}^m, y_m \rangle$ be a sequence of labeled examples where $\tilde{x} \in X$ and $y = \text{sign}(\tilde{u} \cdot \tilde{x})$ for every pair $\langle \tilde{x}, y \rangle \in S$, and let $\mathcal{D}$ be a distribution over S. We will show that the hypothesis h which $\mathtt{WLA}(p, S, \mathcal{D})$ returns is a $(1/2 - \gamma)$-approximator for $\tilde{u}$ under $\mathcal{D}$.

To see that h maps X into $[-1, 1]$, note that for any $\tilde{x} \in X$ Hölder's inequality implies

$$|h(\tilde{x})| = \frac{|\tilde{w} \cdot \tilde{x}|}{\|\tilde{w}\|_q \|X\|_p} \leq \frac{\|\tilde{w}\|_q \|\tilde{x}\|_p}{\|\tilde{w}\|_q \|X\|_p} \leq \frac{\|\tilde{w}\|_q \|X\|_p}{\|\tilde{w}\|_q \|X\|_p} = 1.$$

Now we show that inequality (1) from Section 2.1 holds. Since $h(\tilde{x}^j) \in [-1, 1]$ and $y_j \in \{-1, 1\}$ we have that

$$|h(\tilde{x}^j) - y_j| = 1 - y_j h(\tilde{x}^j),$$

and thus

$$\begin{aligned}
&\frac{1}{2} \sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) |h(\tilde{x}^j) - y_j| \\
&= \frac{1}{2} \sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j)(1 - y_j h(\tilde{x}^j)) \\
&= \frac{1}{2} - \frac{1}{2\|X\|_p} \left(\frac{\sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) y_j (\tilde{w} \cdot \tilde{x}^j)}{\|\tilde{w}\|_q} \right).
\end{aligned}$$

Thus it suffices to show that

$$\frac{\sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) y_j (\tilde{w} \cdot \tilde{x}^j)}{\|\tilde{w}\|_q} \geq \frac{\delta_{\tilde{u},X}}{\|\tilde{u}\|_q}.$$

We first note that

$$\begin{aligned}
\sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) y_j (\tilde{w} \cdot \tilde{x}^j) &= \tilde{w} \cdot \left(\sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) y_j \tilde{x}^j \right) \\
&= \tilde{w} \cdot \tilde{z} = \sum_{j=1}^{m} |z_j|^p = \|\tilde{z}\|_p^p,
\end{aligned}$$

and hence the left-hand side of the desired inequality equals $\|\tilde{z}\|_p^p / \|\tilde{w}\|_q$. We also have

$$\begin{aligned}
\|\tilde{w}\|_q = \left(\sum_{i=1}^{n} \left(|z_i|^{p-1} \right)^q \right)^{1/q} &= \left(\sum_{i=1}^{n} |z_i|^p \right)^{1/q} \\
&= \|\tilde{z}\|_p^{p/q},
\end{aligned}$$

where in the second equality we used the fact that $(p-1)q = p$. Consequently the left-hand side can be further simplified to $\|\tilde{z}\|_p^p / \|\tilde{w}\|_q = \|\tilde{z}\|_p^{p-p/q} = \|\tilde{z}\|_p$, and thus our goal is to show that $\|\tilde{z}\|_p \geq \delta_{\tilde{u},X} / \|\tilde{u}\|_q$. Since $\delta_{\tilde{u},X} \leq \tilde{u} \cdot (y_j \tilde{x}^j)$ for $j = 1, \ldots, m$, we have

$$\begin{aligned}
\delta_{\tilde{u},X} \leq \sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) \tilde{u} \cdot (y_j \tilde{x}^j) &= \tilde{u} \cdot \left(\sum_{j=1}^{m} \mathcal{D}(\tilde{x}^j) y_j \tilde{x}^j \right) \\
&= \tilde{u} \cdot \tilde{z} \\
&\leq \|\tilde{u}\|_q \|\tilde{z}\|_p,
\end{aligned}$$

where the last line follows from the Hölder inequality, and the theorem is proved. ■

4 FROM WEAK TO STRONG LEARNING

We have shown that the simple WLA algorithm is a weak learning algorithm for our halfspace learning problem. In this section we use techniques from boosting and large margin classification to obtain a strong learning algorithm with small sample complexity.

4.1 BOOSTING TO ACHIEVE HIGH ACCURACY

In a series of important papers Schapire [31] and Freund [10, 11] have given *boosting* algorithms which transform weak learning algorithms into strong ones. In this paper we use the Adaboost algorithm from [13] which is shown in Figure 3; our notation for the algorithm is similar to that of [34, 35]. The input to Adaboost is a sequence $S = \langle x^1, y_1 \rangle, \ldots, \langle x^m, y_m \rangle$ of m labeled examples, a weak learning algorithm WL, and two parameters $0 < \gamma, \mu < 1/2$. Given a distribution $\mathcal{D}^t$ over S, algorithm WL outputs a hypothesis $h_t : S \rightarrow [-1, 1]$. Adaboost successively generates new distributions $\mathcal{D}^t$ over S, uses WL to obtain hypotheses h_t, and ultimately outputs as its final hypothesis a linear threshold function over the h_ts.

In [13] Freund and Schapire prove that if the algorithm WL is a $(1/2 - \gamma)$-weak learning algorithm (i.e. each call of WL in Adaboost generates a hypothesis h_t such that $\epsilon_t \leq 1/2 - \gamma$), then the fraction of examples in S which are misclassified by the final hypothesis h is at most μ. Given this result, one straightforward way to obtain a strong learning algorithm for our halfspace learning problem is to draw a sufficiently large (as specified below) sample S from the example oracle $EX(\tilde{u}, \mathcal{D})$ and run Adaboost on S using WLA as the weak learning algorithm, γ as given in Theorem 3, and $\mu < 1/|S|$. This choice of μ ensures that Adaboost's final hypothesis makes no errors on S; moreover, since each hypothesis generated by WLA is of the form $h_t(\tilde{x}) = \tilde{v}^t \cdot \tilde{x}$ for some $\tilde{v}^t \in \Re^n$, Adaboost's final hypothesis is of the form $h(\tilde{x}) = \text{sign}(\tilde{v} \cdot \tilde{x})$ for some $\tilde{v} \in \Re^n$. Using the well-known fact that the VC dimension of the class of zero-bias

```
Input parameters: sequence $S = \langle x^1, y_1\rangle, \ldots, \langle x^m, y_m\rangle$ of labeled examples, weak learning algorithm WL: $S \rightarrow [-1,1]$, two real values $0 < \gamma, \mu < 1/2$

set $T = \frac{1}{2\gamma^2} \log \frac{1}{\mu}$
for all $i = 1, \ldots, m$ set $\mathcal{D}^1(x^i) = \frac{1}{m}$
for $t = 1, \ldots, T$ do
    let $h_t$ be the output of WL($\mathcal{D}^t, S$)
    set $\epsilon_t = \frac{1}{2}\sum_{i=1}^m \mathcal{D}^t(x^i)|h_t(x^i) - y_i|$
    set $\alpha_t = \frac{1}{2}\ln((1-\epsilon_t)/\epsilon_t)$
    for all $i = 1, \ldots, m$ set
        $\mathcal{D}^{t+1}(x^i) = \frac{\mathcal{D}^t(x^i)\exp(-y_i\alpha_t h_t(x^i))}{Z_t}$
    where $Z_t = \sum_{i=1}^m \mathcal{D}^t(x^i)\exp(-y_i\alpha_t h_t(x^i))$ is a normalizing factor (so that $\mathcal{D}^{t+1}$ will be a distribution)
enddo
output as final hypothesis $h(x) \equiv \mathrm{sign}(f(x))$, where
    $f(x) = \frac{\sum_{t=1}^T \alpha_t h_t(x)}{\sum_{t=1}^T \alpha_t}$.
```

Figure 3: The Adaboost algorithm.

linear threshold functions over $\Re^n$ is n, the main result of [7] implies that with probability at least $1-\delta$ the final hypothesis h is an ϵ-accurate hypothesis for $\tilde{u}$ under $\mathcal{D}$ provided that $|S| \geq c(\epsilon^{-1}(n\log(\epsilon^{-1}) + \log(\delta^{-1})))$ for some constant $c > 0$.

This analysis, though attractively simple, yields a rather crude bound on sample complexity which does not depend on the particulars of the learning problem (i.e. $\tilde{u}$ and X). In the rest of this section we use recent results on Adaboost's ability to generate a large-margin classifier and the generalization ability of large-margin classifiers to give a much tighter bound on sample complexity for this learning algorithm.

4.2 BOOSTING TO ACHIEVE A LARGE MARGIN

Suppose that $h : X \rightarrow \{-1,1\}$ is a classifier of the form $h(x) = \mathrm{sign}(f(x))$, where f maps X into $[-1,1]$. We say that the *margin* of h on a labeled example $\langle x, y\rangle$ is $yf(x)$; note that this quantity is nonnegative if and only if h correctly predicts the label y associated with x.

The following theorem, which is an extension of Theorem 5 from [34], shows that Adaboost generates large-margin hypotheses.

Theorem 4 *Suppose that* Adaboost *is run on an example sequence $S = \langle x^1, y_1\rangle, \ldots, \langle x^m, y_m\rangle$ using a weak learning algorithm* WL: $S \rightarrow [-1,1]$. *Then for any value $\theta \geq 0$ we have*

$$\frac{|\{i \in \{1,2,\ldots,m\} : y_i f(x^i) \leq \theta\}|}{m} \leq 2^T \prod_{t=1}^T \sqrt{\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}}.$$

The theorem stated in [34] only covers the case when WL maps S to $\{-1,1\}$. We need this more general version because the weak hypotheses of Theorem 3 map S to $[-1,1]$ rather than $\{-1,1\}$. The proof of Theorem 4 is given in Appendix A.

The results of Section 3 imply that if WLA is used as the weak learning algorithm in Adaboost, then the value ϵ_t will always be at most $1/2-\gamma$, and the upper bound of Theorem 4 becomes $((1-2\gamma)^{1-\theta}(1+2\gamma)^{1+\theta})^{T/2}$. The following easy lemma is proved in Appendix B:

Lemma 5 $(1-4x)^{1-x}(1+4x)^{1+x} \leq 1-4x^2$ *for* $0 \leq x \leq 1/4$.

If we set $\theta = \gamma/2$ and apply this lemma with $x = \theta$, the upper bound of Theorem 4 becomes $(1-\gamma^2)^{T/2}$ and we obtain the following:

Corollary 6 *If* Adaboost *is run on a sequence S of labeled examples drawn from $EX(\tilde{u}, \mathcal{D})$ using* WLA *as the weak learner, γ as defined in Theorem 3 and $\mu < 1/|S|^4$, then the hypothesis h which* Adaboost *generates will have margin at least $\gamma/2$ on every example in S.*

Proof: The bound on μ causes T to be greater than $\frac{2}{\gamma^2}\log\frac{1}{|S|}$, and consequently the upper bound of Theorem 4 is less than $1/|S|$. ■

In the next subsection we use Corollary 6 and the theory of large margin classification to establish a bound on the generalization error of h in terms of the sample size m.

4.3 LARGE MARGINS AND GENERALIZATION ERROR

Let $\mathcal{F}$ be a collection of real-valued functions on a set X. A finite set $\{x^1, \ldots, x^k\} \subseteq X$ is said to be ξ*-shattered by*

$\mathcal{F}$ if there are real numbers $r_1, \ldots, r_k$ such that for all $b = (b_1, \ldots, b_k) \in \{-1,1\}^k$, there is a function $f_b \in \mathcal{F}$ such that

$$f_b(x^i) \begin{cases} \geq r_i + \xi & \text{if } b_i = 1 \\ \leq r_i - \xi & \text{if } b_i = -1. \end{cases}$$

For $\xi \geq 0$, the *fat-shattering dimension of $\mathcal{F}$ at scale ξ*, denoted $\text{fat}_{\mathcal{F}}(\xi)$, is the size of the largest set which is ξ-shattered by $\mathcal{F}$, if this is finite, and infinity otherwise. The fat-shattering dimension is useful for us because of the following theorem from [4]:

Theorem 7 *Let $\mathcal{F}$ be a collection of real-valued functions on X and let $\mathcal{D}$ be a distribution over $X \times \{-1,1\}$. Let $S = \langle \tilde{x}^1, y_1 \rangle, \ldots, \langle \tilde{x}^m, y_m \rangle$ be a sequence of labeled examples drawn from $\mathcal{D}$. With probability at least $1-\delta$ over the choice of S, if a classifier $h(x) \equiv \text{sign}(f(x))$ with $f \in \mathcal{F}$ has margin at least $\xi > 0$ on every example in S, then*

$$\Pr_{(x,y)\in\mathcal{D}}[h(x) \neq y] \leq \frac{2}{m}\left(d \log \frac{8em}{d} \log(32m) + \log \frac{8m}{\delta}\right),$$

where $d = \text{fat}_{\mathcal{F}}(\xi/16)$.

As noted in Section 4.1, the final hypothesis h which `Adaboost` outputs must be of the form $h(\tilde{x}) = \text{sign}(f(\tilde{x}))$ with $f(\tilde{x}) = \tilde{v} \cdot \tilde{x}$ for some $\tilde{v} \in \Re^n$. Furthermore, since each invocation of `WLA` generates a hypothesis of the form $h_t(\tilde{x}) = \tilde{v}^t \cdot \tilde{x}$ with $\|\tilde{v}^t\|_q \leq \frac{1}{\|X\|_p}$, Minkowski's inequality implies that the vector $\tilde{v}$ must satisfy $\|\tilde{v}\|_q \leq \frac{1}{\|X\|_p}$. We thus consider the class of functions

$$\mathcal{F} = \left\{ \tilde{x} \mapsto \tilde{v} \cdot \tilde{x} : \|\tilde{v}\|_q \leq \frac{1}{\|X\|_p}, \|\tilde{x}\|_p \leq \|X\|_p \right\}. \qquad (2)$$

If we can bound $\text{fat}_{\mathcal{F}}(\xi)$, then given any sample size m, Theorem 7 immediately yields a corresponding bound on $\Pr_{x\in\mathcal{D}}[h(\tilde{x}) \neq \text{sign}(\tilde{u} \cdot \tilde{x})]$ for our halfspace learning problem. The following theorem proved in Appendix C gives the desired bound on $\text{fat}_{\mathcal{F}}(\xi)$:

Theorem 8 *Let X be a bounded region in $\Re^n$ and let $\mathcal{F}$ be the class of functions on X defined in (2) above. Then $\text{fat}_{\mathcal{F}}(\xi) \leq \frac{2 \log 4n}{\xi^2}$.*

Combining Theorem 3, Corollary 6, and Theorems 7 and 8, it follows that if our algorithm uses a sample of size $|S| = m$, then with probability at least $1-\delta$ the hypothesis h which is generated will satisfy

$$\begin{aligned} &\Pr_{\tilde{x}\in\mathcal{D}}[h(\tilde{x}) \neq \text{sign}(\tilde{u} \cdot \tilde{x})] \\ &\quad = O\left(\frac{1}{m}\left(\frac{\|\tilde{u}\|_q^2 \|X\|_p^2}{\delta_{\tilde{u},X}^2} \log n \log^2 m + \log \frac{m}{\delta}\right)\right). \end{aligned}$$

Thus we have established the following (where the $\tilde{O}$-notation hides log factors):

Theorem 9 *The algorithm obtained by applying* `Adaboost` *to* `WLA` *using the parameter settings described in Corollary 6 is a strong learning algorithm for $\tilde{u}$ on X with sample complexity*

$$m(\epsilon, \delta, \tilde{u}, X) = \tilde{O}\left(\frac{1}{\epsilon} \cdot \frac{\|\tilde{u}\|_q^2 \|X\|_p^2}{\delta_{\tilde{u},X}^2}\right).$$

5 DISCUSSION

The sample complexity of our boosting-based p-norm PAC learning algorithm is remarkably similar to that of the PAC-transformed online p-norm algorithms of Section 2.1. Up to log factors both sets of bounds depend linearly on ϵ^{-1} and quadratically on $\|\tilde{u}\|_q \|X\|_p / \delta_{\tilde{u},X}$. Comparing the bounds in more detail, we see that the online variant described in part (a) of Theorem 1 has an extra factor of $p-1$ in its bound which is not present in the sample complexity of our algorithm. Variant (a) offers the advantage, though, that the user does not need to know the values of any quantities such as $\|X\|_p$ or $\|\tilde{u}\|_q$ in advance in order to run the algorithm. Turning to part (b) of Theorem 1, we see that if the parameter a is set appropriately in the online algorithm then the online bound differs from our PAC algorithm bound only by an extra factor of

$$(p-1)\left(1 - \left(\frac{\tilde{u} \cdot \tilde{z}^0}{\|\tilde{u}\|_q \|\tilde{z}^0\|_p}\right)^2\right)$$

(again ignoring log factors). Part (c) of Theorem 1 shows that as $p \to \infty$ this extra factor becomes quite small even when $\tilde{z}^0$ is chosen to be $(1, \ldots, 1)$. We also note that when $p = \Omega(\log n)$ Gentile and Littlestone [15] have given alternative expressions for the online p-norm bounds in terms of $\|X\|_\infty$ and $\|\tilde{u}\|_1$. Using an entirely similar analysis the bounds of our algorithm can be analogously rephrased in this case as well.

5.1 $p = 2$ AND THE PERCEPTRON ALGORITHM

Since the $p = 2$ case of the online p-norm algorithm is precisely the Perceptron algorithm, the $p = 2$ case of our algorithm can be viewed as a natural PAC-model analogue of the online Perceptron algorithm. We note that when $p = 2$ the upper bound given in Lemma 12 of Appendix C can be strengthened to $\sqrt{d} \cdot \|X\|_2$ (see Lemma 1.3 of [4] or Theorem 4.1 of [2] for a proof). This means that the fat-shattering dimension upper bound of Theorem 8 can be improved to $\frac{1}{\xi^2}$, which removes a log factor from the bound of Theorem 9; however this bound will still contain various log factors because of the log terms in Theorem 7.

5.2 $p = \infty$ AND THE JACKSON-CRAVEN ALGORITHM

At the other extreme, we now define a natural $p = \infty$ version of our algorithm. Consider the vectors $\tilde{z}$ and $\tilde{w}$ which are computed by the weak learning algorithm `WLA`. If we let r be the number of coordinates z_i of $\tilde{z}$ such that $|z_i| = \|\tilde{z}\|_\infty$, then for any i we have

$$\begin{aligned} \lim_{p\to\infty}\left(\frac{w_i}{\|\tilde{w}\|_q}\right) &= \lim_{p\to\infty}\left(\frac{\text{sign}(z_i)|z_i|^{p-1}}{(\sum_{i=1}^n |z_i|^{(p-1)q})^{1/q}}\right) \\ &= \begin{cases} \text{sign}(z_i)/r & \text{if } |z_i| = \|\tilde{z}\|_\infty \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

Hence it is natural to consider a $p = \infty$ version of `WLA`, which we denote `WLA′`, in which the vector $\tilde{w}$ is defined by taking $w_i = \text{sign}(z_i)$ if $|z_i| = \|\tilde{z}\|_\infty$ and $w_i = 0$ otherwise. All of our analysis continues to hold (with minor modifications described in Appendix D) and we obtain a $p = \infty$ strong learning algorithm:

Claim 10 *Theorem 9 holds for $p = \infty$ with* $\mathtt{WLA}'$ *in place of* $\mathtt{WLA}$.

There is a close relationship between this $p = \infty$ algorithm and the work of Jackson and Craven on learning sparse perceptrons [20]. Note that if $r = 1$, i.e. only one coordinate of $\tilde{z}$ has $|z_i| = \|\tilde{z}\|_\infty$, then the $\mathtt{WLA}'$ hypothesis is $h(\tilde{x}) = \frac{\ell}{\|X\|_\infty}$ where ℓ is the signed variable from

$$\{x_1, \dots, x_n, -x_1, \dots, -x_n\}$$

which is most strongly correlated under distribution $\mathcal{D}$ with the value of $\text{sign}(\tilde{u} \cdot \tilde{x})$. This is very similar to the weak learning algorithm used by Jackson and Craven in [20], which takes the single best-correlated literal as its hypothesis (breaking ties arbitrarily).

The proof that this "best-single-literal" algorithm used in [20] is a weak learning algorithm is due to Goldmann, Håstad and Razborov [17]. However, the proof in [17] assumes that the example space X is $\{0,1\}^n$ and the target vector $\tilde{u}$ has all integer coefficients; thus, as noted by Jackson and Craven in [20], their algorithm for learning sparse perceptrons only applies to learning problems which are defined over discrete input domains. In contrast, our $p = \infty$ algorithm can be applied on continuous input domains – the only restrictions are that the example space X and the target vector $\tilde{u}$ satisfy $\|X\|_\infty < \infty$ and $\delta_{\tilde{u},X} > 0$.

We also observe that Theorem 9 establishes a tighter sample complexity bound for our $p = \infty$ strong learning algorithm than was given in [20]. To see this, let $X = \{0,1\}^n$ and suppose that the target vector $\tilde{u} \in \Re^n$ has all integer coefficients, so the algorithm from [20] can be applied. For this learning problem we have $\delta_{\tilde{u},X} = \Omega(1)$ and $\|X\|_\infty = 1$; letting $s = \|\tilde{u}\|_1$, Theorem 9 implies that our $p = \infty$ strong learning algorithm has sample complexity roughly s^2/ϵ (ignoring log factors). This is a substantial improvement over the roughly s^4/ϵ sample complexity bound given in [20]. More generally, the sample complexity bound given in [20] for learning "s-sparse k-perceptrons" is roughly ks^4/ϵ; the analysis of this paper can easily be extended to establish a sample complexity bound of roughly ks^2/ϵ for learning s-sparse k-perceptrons.

6 OPEN QUESTIONS

Our results give evidence of the broad utility of boosting algorithms such as Adaboost. A natural question is how much further this utility extends: are there simple boosting-based PAC versions of other standard learning algorithms? We note in this context that Kearns and Mansour [23] have shown that various heuristic algorithms for top-down decision tree induction can be viewed as instantiations of boosting. Another goal is to construct more powerful boosting-based PAC algorithms for linear threshold functions. All of the algorithms discussed in this paper have an inverse quadratic dependence on the separation parameter $\delta_{\tilde{u},X}$; linear-programming based algorithms for learning linear threshold functions (see, e.g., [6, 7, 9, 29, 30]) do not have such a dependence. Is there a natural boosting-based PAC algorithm for linear threshold functions with performance bounds similar to those of the linear-programming based algorithms?

7 ACKNOWLEDGEMENTS

We warmly thank Les Valiant for helpful comments and suggestions.

References

[1] D. Angluin. Queries and concept learning, *Machine Learning* **2** (1988), 319-342.

[2] N. Alon, J. Spencer and P. Erdos. *The Probabilistic Method*, Wiley-Interscience, New York, 1992.

[3] P. Auer and M. Warmuth. Tracking the best disjunction, *in* "Proc. 36th Symp. on Found. of Comp. Sci." (1995), 312-321.

[4] P. Bartlett and J. Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers, *in* B. Scholkopf, C.J.C. Burges, and A.J. Smola, editors, "Advances in Kernel Methods – Support Vector Learning," Cambridge, MA, MIT Press (1999), 43-54.

[5] E. Baum. The perceptron algorithm is fast for nonmalicious distributions, *Neural Computation* **2**(2) (1990), 248-260.

[6] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial time algorithm for learning noisy linear threshold functions, *in* "Proc. 37th Symp. on Found. of Comp. Sci." (1996), 330-338.

[7] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis Dimension, *J. ACM* **36**(4) (1989), 929-965.

[8] T. Bylander. Worst-case analysis of the perceptron and exponentiated update algorithms, *Artificial Intelligence* **106** (1998).

[9] E. Cohen. Learning noisy perceptrons by a perceptron in polynomial time, *in* "Proc. 38th Symp. on Found. of Comp. Sci." (1997), 514-523.

[10] Y. Freund. Boosting a weak learning algorithm by majority, *Inform. and Comp.* **121**(2) (1995), 256-285.

[11] Y. Freund. An improved boosting algorithm and its implications on learning complexity, *in* "Fifth Ann. Work. on Comp. Learning Theory" (1992), 391-398.

[12] Y. Freund and R. Schapire. Game theory, on-line prediction and boosting, *in* "Proc. Ninth Ann. Conf. on Comp. Learning Theory" (1996), 325-332.

[13] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, *J. Comp. System Sci.* **55**(1) (1997), 119-139.

[14] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm, *in* "Proc. Eleventh Ann. Conf. on Comp. Learning Theory" (1998), 209-217.

[15] C. Gentile and N. Littlestone. The robustness of the p-norm algorithms, *in* "Proc. 12th Ann. Conf. on Comp. Learning Theory" (1999), 1-11.

[16] A. Grove, N. Littlestone and D. Schuurmans. General convergence results for linear discriminant updates, *in* "Proc. 10th Ann. Conf. on Comp. Learning Theory" (1997), 171-183.

[17] M. Goldmann, J. Håstad and A. Razborov. Majority gates vs. general weighted threshold gates, *Comput. Complexity* **2** (1992), 277-300.

[18] D. Haussler. Space efficient learning algorithms, Technical Report UCSC-CRL-88-2, University of Calif., Santa Cruz, 1988.

[19] W. Hoeffding. Probability inequalities for sums of bounded random variables, *J. Amer. Statist. Assoc* **58** (1963), 13-30.

[20] J. Jackson and M. Craven. Learning sparse perceptrons, *in* "Advances in Neural Information Processing Systems 8," MIT Press (1996), 654-660.

[21] C. Ji and S. Ma. Combinations of weak classifiers, *IEEE Trans. Neural Networks* **8**(1) (1997), 32-42.

[22] M. Kearns, M. Li, L. Pitt and L. Valiant. Recent results on boolean concept learning, *in* "Proc. Fourth Int. Workshop on Machine Learning" (1987), 337-352.

[23] M. Kearns and Y. Mansour. On the boosting ability of top-down decision tree learning algorithms, *in* "Proc. 28th Symp. on Theor. of Comp.," (1996), 459-468.

[24] M. Kearns and L. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM*, **41**(1) (1994), 67-95. (Preliminary version appeared in "Proc. 21st ACM Symp. on Theor. of Comp.," (1989), 433-444.)

[25] J. Kivinen, M. Warmuth and P. Auer. The perceptron algorithm versus winnow: linear vs. logarithmic mistake bounds when few input variables are relevant, *in* "Proc. Eighth Ann. Conf. on Comp. Learning Theory" (1995), 289-296.

[26] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm, *Machine Learning* **2** (1988), 285-318.

[27] N. Littlestone. Mistake bounds and logarithmic linear-threshold learning algorithms, Ph.D. thesis, Technical Report UCSC-CRL-89-11, University of Calif., Santa Cruz, 1989.

[28] N. Littlestone. Redundant Noisy attributes, attribute errors, and linear-threshold learning using winnow, *in* "Proc. Fourth Ann. Conf. on Comp. Learning Theory" (1991), 147-156.

[29] P. Long. Halfspace learning, linear programming, and nonmalicious distributions, *Inf. Proc. Let.* **51** (1994), 245-250.

[30] W. Maass and G. Turan. How fast can a threshold gate learn? *in* "Comput. Learning Theory and Natural Learning Systems: Volume I: Constraints and Prospects," S. J. Hanson, G. Drastal, & R. Rivest, eds., MIT Press, Cambridge, MA, 1994, 381-414.

[31] R. Schapire. The strength of weak learnability, *Machine Learning* **5**(2) (1990), 197-227.

[32] R. Schapire. Drifting games, *in* "Proc. Twelfth Ann. Conf. on Comp. Learning Theory" (1999), 114-124.

[33] R. Servedio. On PAC learning using Winnow, Perceptron, and a Perceptron-like algorithm, *in* "Proc. Twelfth Ann. Conf. on Comp. Learning Theory" (1999), 296-307.

[34] R. Schapire, Y. Freund, P. Bartlett and W.S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods, *Annals of Statistics* **26**(5) (1998), 1651-1686.

[35] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions, *in* "Proc. Eleventh Ann. Conf. on Comp. Learning Theory" (1998), 80-91.

[36] M. Schmitt. Identification criteria and lower bounds for Perceptron-like learning rules, *Neural Computation* **10** (1998), 235-250.

[37] A. Taylor and W. Mann. *Advanced Calculus*, Wiley & Sons, 1972.

A PROOF OF THEOREM 4

The proof combines ideas from [34], where it is shown that `Adaboost` with binary valued hypotheses generates a large margin classifier, and [35], where an analysis is given for `Adaboost` with real valued hypotheses. As in Theorem 5 of [34], if $y_i f(x^i) \le \theta$ then

$$y_i \sum_{t=1}^{T} \alpha_t h_t(x^i) \le \theta \sum_{t=1}^{T} \alpha_t$$

which implies that

$$\exp\left(-y_i \sum_{t=1}^{T} \alpha_t h_t(x^i) + \theta \sum_{t=1}^{T} \alpha_t\right) \ge 1.$$

Following [34], we thus have

$$\begin{aligned}
&\frac{|\{i \in \{1,2,\ldots,m\} : y_i f(x^i) \le \theta\}|}{m} \\
\le\ & \sum_{i=1}^{m} \frac{1}{m} \cdot \left[\exp\left(-y_i \sum_{t=1}^{T} \alpha_t h_t(x^i) + \theta \sum_{t=1}^{T} \alpha_t\right)\right] \\
=\ & \frac{\exp\left(\theta \sum_{t=1}^{T} \alpha_t\right)}{m} \sum_{i=1}^{m} \exp\left(-y_i \sum_{t=1}^{T} \alpha_t h_t(x^i)\right) \\
=\ & \exp\left(\theta \sum_{t=1}^{T} \alpha_t\right) \left(\prod_{t=1}^{T} Z_t\right) \sum_{i=1}^{m} \mathcal{D}^{T+1}(x^i) \\
=\ & \exp\left(\theta \sum_{t=1}^{T} \alpha_t\right) \left(\prod_{t=1}^{T} Z_t\right) \qquad (3)
\end{aligned}$$

where the second equality follows from the definition of $\mathcal{D}^{t+1}$ and the final equality is because $\mathcal{D}^{T+1}$ is a distribution and hence sums to 1. Our goal is thus to bound the right side of inequality (3).

If we let

$$r_t = \sum_{i=1}^{m} \mathcal{D}^t(x^i) y_i h_t(x^i)$$

then using the fact that

$$|h(x^j) - y_j| = 1 - y_j h(x^j)$$

we find that $\epsilon_t = \frac{1-r_t}{2}$. Substituting into the definition of α_t we obtain

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1+r_t}{1-r_t}\right).$$

Following [35] for simplicity of notation we now fix t and let $u_i = y_i h_t(x^i)$, $Z = Z_t$, $\mathcal{D} = \mathcal{D}^t$, $\epsilon = \epsilon_t$, $r = r_t$, and $\alpha = \alpha_t$. As noted in [35] a simple convexity argument shows that

$$e^{-\alpha u} \le \frac{1+u}{2} e^{-\alpha} + \frac{1-u}{2} e^{\alpha}$$

for any $\alpha \in \Re$ and any $u \in [-1,1]$. Since u_i always lies in the interval $[-1,1]$, we can apply this inequality to obtain

$$\begin{aligned} Z &= \sum_{i=1}^{m} \mathcal{D}(x^i) e^{-\alpha u_i} \\ &\le \sum_{i=1}^{m} \mathcal{D}(x^i) \left(\frac{1+u_i}{2} e^{-\alpha} + \frac{1-u_i}{2} e^{\alpha} \right). \end{aligned} \tag{4}$$

As in Section 3.5 of [35], substituting α into inequality (4) yields

$$\begin{aligned} Z_t &\le \sqrt{1-r_t^2} \\ &= \sqrt{1-(1-2\epsilon_t)^2} \\ &= 2\sqrt{\epsilon_t(1-\epsilon_t)}. \end{aligned} \tag{5}$$

Substituting inequality (5) into inequality (3) and using the definition of α_t yields the desired bound of the theorem. ■

B PROOF OF LEMMA 5

We show that

$$(1-4x)^{1-x}(1+4x)^{1+x} \le 1-4x^2$$

for $0 \le x \le 1/4$. Using a simple convexity argument, it can be verified that $\alpha^r \le 1-(1-\alpha)r$ for any $\alpha \ge 0$ and any $0 \le r \le 1$. This inequality implies that $(1-4x)^{1-x} \le 1-4x+4x^2$ and $(1+4x)^x \le 1+4x^2$, so consequently $(1-4x)^{1-x}(1+4x)^{1+x} \le (1-4x+4x^2)(1+4x)(1+4x^2)$, which is at most $1-4x^2$ for $0 \le x \le 1/4$. ■

C PROOF OF THEOREM 8

The theorem is a variant of Theorem 1.6 from [4]. The proof follows from combining the inequalities proved in the following two lemmas.

Lemma 11 *Let*

$$\mathcal{F} = \left\{ \tilde{x} \mapsto \tilde{v} \cdot \tilde{x} : \|\tilde{v}\|_q \le \frac{1}{\|X\|_p}, \|\tilde{x}\|_p \le \|X\|_p \right\}.$$

If the set $\{\tilde{x}^1, \ldots, \tilde{x}^d\}$ is ξ-shattered by $\mathcal{F}$ then every $b = (b_1, \ldots, b_d) \in \{-1,1\}^d$ satisfies $\left\|\sum_{i=1}^d b_i \tilde{x}^i\right\|_p \ge \xi d \|X\|_p$.

Proof: Suppose that $\{\tilde{x}^1, \ldots, \tilde{x}^d\}$ is ξ-shattered by $\mathcal{F}$ as witnessed by the real numbers $r_1, \ldots, r_d$. Then for every $b = (b_1, \ldots, b_d) \in \{-1,1\}^d$, there is a vector $\tilde{v}_b \in \Re^n$ with $\|\tilde{v}_b\|_q \le \frac{1}{\|X\|_p}$ such that $b_i(\tilde{v}_b \cdot \tilde{x}^i - r_i) \ge \xi$ for $i = 1, \ldots, d$. Summing these d inequalities and rearranging, we obtain

$$\tilde{v}_b \cdot \left(\sum_{i=1}^d b_i \tilde{x}^i \right) \ge \xi d + \sum_{i=1}^d b_i r_i. \tag{6}$$

There are two cases to consider. Case 1 is if $\sum_{i=1}^d b_i r_i \ge 0$; if this is true, we have

$$\begin{aligned} \frac{1}{\|X\|_p} \cdot \left\| \sum_{i=1}^d b_i \tilde{x}^i \right\|_p &\ge \|\tilde{v}_b\|_q \left\| \sum_{i=1}^d b_i \tilde{x}^i \right\|_p \\ &\ge \tilde{v}_b \cdot \left(\sum_{i=1}^d b_i \tilde{x}^i \right) \\ &\ge \xi d \end{aligned}$$

(where the first inequality is by the definition of $\mathcal{F}$, the second inequality is Hölder's, and the third is from inequality (6)), which yields the desired inequality $\|\sum_{i=1}^d b_i \tilde{x}^i\|_p \ge \xi d \|X\|_p$.

In the second case, $\sum_{i=1}^d b_i r_i < 0$. If this is the case then let $c = (c_1, \ldots, c_d) = (-b_1, \ldots, -b_d)$. We then have $\sum_{i=1}^d c_i r_i > 0$, so Case 1 implies that $\|\sum_{i=1}^d c_i \tilde{x}^i\|_p \ge \xi d \|X\|_p$, and the lemma follows since

$$\left\| \sum_{i=1}^d c_i \tilde{x}^i \right\|_p = \left\| - \sum_{i=1}^d b_i \tilde{x}^i \right\|_p = \left\| \sum_{i=1}^d b_i \tilde{x}^i \right\|_p.$$

■

Lemma 12 *For any set $\{\tilde{x}^1, \ldots, \tilde{x}^d\}$ with each $\|\tilde{x}^i\|_p \le \|X\|_p$, if $p \ge 2$ then there is some $b = (b_1, \ldots, b_d) \in \{-1,1\}^d$ such that $\left\|\sum_{i=1}^d b_i \tilde{x}^i\right\|_p \le \sqrt{2d \log 4n} \cdot \|X\|_p$.*

Proof: The proof uses the probabilistic method. We consider the random variable $\tilde{z} = \sum_{i=1}^d b_i \tilde{x}^i$ where $(b_1, \ldots, b_d)$ is uniformly distributed over $\{-1,1\}^d$. For any coordinate $j \in \{1, \ldots, n\}$ we have $z_j = \sum_{i=1}^d b_i x_j^i$ and hence $E[z_j] = 0$. Let $Y_j = |x_j^1|^2 + \cdots + |x_j^d|^2$; Hoeffding's bound [19] on sums of independent random variables states that for any $t > 0$ we have

$$\Pr[|z_j| > t] \le 2 \exp\left(\frac{-t^2}{2Y_j} \right).$$

As a consequence, taking $t = \sqrt{2Y_j \log 4n}$ we have that $\Pr[|z_j| \ge t] \le 1/2n$. Using the union bound across $j = 1, 2, \ldots, n$, we have that with probability at least $1/2$ every coordinate z_j of $\tilde{z}$ satisfies $|z_j| < \sqrt{2Y_j \log 4n}$, and hence

$$\begin{aligned} \|\tilde{z}\|_p &= \left(\sum_{j=1}^n |z_j|^p \right)^{1/p} \\ &\le \left(\sum_{j=1}^n \left(\sqrt{2Y_j \log 4n} \right)^p \right)^{1/p} \\ &= \sqrt{2 \log 4n} \cdot \\ &\quad \left(\left(\sum_{j=1}^n \left[|x_j^1|^2 + \cdots + |x_j^d|^2 \right]^{p/2} \right)^{2/p} \right)^{1/2} \end{aligned} \tag{7}$$

Since $p \ge 2$, we have $p/2 \ge 1$ and hence Minkowski's inequality implies that

$$\begin{aligned} &\left(\sum_{j=1}^n \left[|x_j^1|^2 + \cdots + |x_j^d|^2 \right]^{p/2} \right)^{2/p} \\ &\le \left[\sum_{j=1}^n |x_j^1|^{2p/2} \right]^{2/p} + \cdots + \left[\sum_{j=1}^n |x_j^d|^{2p/2} \right]^{2/p} \\ &= \|\tilde{x}^1\|_p^2 + \cdots + \|\tilde{x}^d\|_p^2 \\ &\le d \|X\|_p^2. \end{aligned} \tag{8}$$

The lemma follows by combining inequalities (7) and (8). ■

D PROOF OF CLAIM 10

We first show that Theorem 3 still holds in the case $p = \infty$ with $\mathtt{WLA}'$ in place of $\mathtt{WLA}$. The proof is unchanged up through the point where we must show that

$$\frac{\sum_{i=1}^{m} \mathcal{D}(\tilde{x}^i) y_i (\tilde{w} \cdot \tilde{x}^i)}{\|\tilde{w}\|_1} \geq \frac{\delta_{\tilde{u},X}}{\|\tilde{u}\|_1}.$$

The left-hand side of this inequality can be rewritten as

$$\begin{aligned} \frac{\tilde{w} \cdot \tilde{z}}{\|\tilde{w}\|_1} &= \frac{\sum_{|z_i| = \|\tilde{z}\|_\infty} \operatorname{sign}(z_i) z_i}{\sum_{|z_i| = \|\tilde{z}\|_\infty} 1} \\ &= \frac{\sum_{|z_i| = \|\tilde{z}\|_\infty} \|\tilde{z}\|_\infty}{\sum_{|z_i| = \|\tilde{z}\|_\infty} 1} \\ &= \|\tilde{z}\|_\infty, \end{aligned}$$

and hence it suffices to prove that $\|\tilde{z}\|_\infty \geq \delta_{\tilde{u},X} / \|\tilde{u}\|_1$. This is established at the end of the proof of Theorem 3, so Theorem 3 holds with $p = \infty$ and $\mathtt{WLA}'$ substituted for $\mathtt{WLA}$.

The rest of the analysis goes through unchanged except for inequalities (7) and (8) of Lemma 12. Since $\|X\|_\infty = \sup_{\tilde{x} \in X} \max_{j=1,\ldots,n} |x_j|$, we have that $Y_j \leq d\|X\|_\infty^2$ for all j, and hence in place of inequalities (7) and (8) we have

$$\begin{aligned} \|\tilde{z}\|_\infty = \max_j |z_j| &\leq \max_j \sqrt{2 Y_j \log 4n} \\ &\leq \sqrt{2d \log 4n} \cdot \|X\|_\infty, \end{aligned}$$

which proves the lemma. ■