Journal of Machine Learning Research 10 (2009) 281-299 Submitted 12/07; 12/08; Published 2/09

Analysis of Perceptron-Based Active Learning

Sanjoy Dasgupta DASGUPTA@CS.UCSD.EDU
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0404, USA

Adam Tauman Kalai ADUM@MICROSOFT.COM
Microsoft Research
Office 14063
One Memorial Drive
Cambridge, MA 02142, USA

Claire Monteleoni CMONTEL@CCLS.COLUMBIA.EDU
Center for Computational Learning Systems
Columbia University
Suite 850, 475 Riverside Drive, MC 7717
New York, NY 10115, USA

Editor: Manfred Warmuth

Abstract

We start by showing that in an active learning setting, the Perceptron algorithm needs $\Omega(\frac{1}{\varepsilon^2})$ labels to learn linear separators within generalization error ε. We then present a simple active learning algorithm for this problem, which combines a modification of the Perceptron update with an adaptive filtering rule for deciding which points to query. For data distributed uniformly over the unit sphere, we show that our algorithm reaches generalization error ε after asking for just $\tilde{O}(d \log \frac{1}{\varepsilon})$ labels. This exponential improvement over the usual sample complexity of supervised learning had previously been demonstrated only for the computationally more complex query-by-committee algorithm.

Keywords: active learning, perceptron, label complexity bounds, online learning

1. Introduction

In many machine learning applications, unlabeled data is abundant but labeling is expensive. This distinction is not captured in standard models of supervised learning, and has motivated the field of *active learning*, in which the labels of data points are initially hidden, and the learner must pay for each label it wishes revealed. If query points are chosen randomly, the number of labels needed to reach a target generalization error ε, at a target confidence level $1-\delta$, is similar to the sample complexity of supervised learning. The hope is that there are alternative querying strategies which require significantly fewer labels.

An early dramatic demonstration of the potential of active learning was Freund et al.'s analysis of the query-by-committee (QBC) learning algorithm (Freund et al., 1997). The analysis is with respect to the *selective sampling* model: the learner observes a stream of unlabeled data and makes spot decisions about whether or not to ask for each point's label. The paper showed that if the data is drawn uniformly from the surface of the unit sphere in $\mathbb{R}^d$, and the hidden labels correspond

perfectly to a homogeneous (i.e., through the origin) linear separator from this same distribution, then it is possible to achieve generalization error ε after seeing $\tilde{O}(\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$ points and requesting just $\tilde{O}(d\log\frac{1}{\varepsilon})$ labels:[1] an exponential improvement over the usual $\tilde{O}(\frac{d}{\varepsilon})$ sample complexity of learning linear separators in a supervised setting. (An $\Omega(d\log\frac{1}{\varepsilon})$ label complexity can be seen to be optimal by counting the number of spherical caps of radius ε that can be packed onto the surface of the unit sphere in $\mathbb{R}^d$.) This remarkable result is tempered somewhat by the complexity of the QBC algorithm, which involves random sampling from intermediate version spaces; the complexity of the update step scales (polynomially) with the number of updates performed.

In this paper, we show how a simple modification of the perceptron update can be used to achieve the same sample complexity bounds (within $\tilde{O}$ factors), under the same streaming model and the same uniform input distribution. Unlike QBC, we do not assume a distribution over target hypotheses, and our algorithm does not need to store previously seen data points, only its current hypothesis. Moreover, in addition to requiring only one-at-a-time access to examples (as opposed to batch data access), neither our algorithm's memory usage, nor its computation time per example, scales with the number of seen examples.[2]

Our algorithm has the following structure.

```
Set initial hypothesis v_0 ∈ R^d.
For t = 0,1,2,....
  Receive unlabeled point x_t.
  Make a prediction SGN(v_t · x_t).
  Filtering step: Decide whether to ask for x_t's label.
  If label y_t is requested:
    Update step: Set v_{t+1} based on v_t, x_t, y_t.
    Adjust filtering rule.
  else: v_{t+1} = v_t.
```

UPDATE STEP.

The regular perceptron update, whose convergence behavior was first analyzed by Rosenblatt (1958), consists of the following simple rule:

$$\text{if } (x_t, y_t) \text{ is misclassified, then } v_{t+1} = v_t + y_t x_t.$$

It turns out that this update cannot yield an error rate better than $\Omega(1/\sqrt{l_t})$, where l_t is the number of labels queried up to time t, no matter what filtering scheme is used.

Theorem 1 *Consider any sequence of data points $x_0, x_1, x_2, \ldots \in \mathbb{R}^d$ which is perfectly classified by some linear separator $u \in \mathbb{R}^d$. Suppose that perceptron updates are used, starting with an initial hypothesis v_0. Let k_t be the number of updates performed upto time t, let v_t be the hypothesis at time t, and let θ_t be the angle between u and v_t. Then for any $t \geq 0$, if $\theta_{t+1} \leq \theta_t$ then $\sin\theta_t \geq 1/(5\sqrt{k_t + \|v_0\|^2})$.*

This holds regardless of how the data is produced. When the points are distributed uniformly over the unit sphere, $\theta_t \geq \sin\theta_t$ (for $\theta_t \leq \frac{\pi}{2}$) is proportional to the error rate of v_t. In other words, the

1. In this paper, the $\tilde{O}$ notation is used to suppress multiplicative terms in $\log d, \log\log\frac{1}{\varepsilon}$ and $\log\frac{1}{\delta}$.
2. See Monteleoni (2006) for an extended discussion of learning with online constraints.

error rate is $\Omega(1/\sqrt{k_t})$, which in turn is $\Omega(1/\sqrt{l_t})$, since each update must be triggered by a label. As we will shortly see, the reason for this slow rate is that the magnitude of the perceptron update is too large for points near the decision boundary of the current hypothesis.

So instead we use a *variant* of the update rule, originally due to Motzkin and Schoenberg (1954):

$$\text{if } (x_t, y_t) \text{ is misclassified, then } v_{t+1} = v_t - 2(v_t \cdot x_t)x_t$$

(where x_t is assumed normalized to unit length). Note that the update can also be written as $v_{t+1} = v_t + 2y_t|v_t \cdot x_t|x_t$, since updates are only made on mistakes, in which case $y_t \neq \mathtt{SGN}(v_t \cdot x_t)$, by definition. Thus we are scaling the standard perceptron's additive update by a factor of $2|v_t \cdot x_t|$ to avoid oscillations caused by points close to the half-space represented by the current hypothesis. Motzkin and Schoenberg (1954) introduced this rule, in the context of solving linear inequalities, and called it the "Reflexion" method, due to certain geometric properties it has, which we will discuss later. Hampson and Kibler (1999) subsequently applied it to learning linear separators, in an analysis framework that differs from ours. The same rule, but without the factor of two, has been used in previous work (Blum et al., 1996) on learning linear classifiers from noisy data, in a batch setting. We are able to show that our formulation has the following generalization performance in a supervised (non-active) setting.

Theorem 2 *Pick any $\delta, \varepsilon > 0$. Consider a stream of data points x_t drawn uniformly at random from the surface of the unit sphere in $\mathbb{R}^d$, and corresponding labels y_t that are consistent with some linear separator. When the modified Perceptron algorithm (Figure 2) is applied to this stream of data, then with probability $1-\delta$, after $O(d(\log\frac{1}{\varepsilon} + \log\frac{1}{\delta}))$ mistakes, its generalization error is at most ε.*

This contrasts favorably with the $\tilde{O}(\frac{d}{\varepsilon^2})$ mistake bound of the Perceptron algorithm, and a more recent variant, on the same distribution (Baum, 1997; Servedio, 1999). Meanwhile, in terms of lower bounds, Theorem 1 also applies in the supervised case, and gives a lower bound on the number of mistakes (updates) made by the standard perceptron. Finally, there is the question of how many samples are needed in the supervised setting (as opposed to the number of mistakes). For data distributed uniformly over the unit sphere, this is known to be $\tilde{\Theta}(\frac{d}{\varepsilon})$ (lower bound, Long, 1995, and upper bound, Long, 2003).

FILTERING STEP.

Given the limited information the algorithm keeps, a natural filtering rule is to query points x_t when $|v_t \cdot x_t|$ is less than some threshold s_t. The choice of s_t is crucial. If it is too large, then only a miniscule fraction of the points queried will actually be misclassified (and thus trigger updates)—almost all labels will be wasted. On the other hand, if s_t is too small, then the waiting time for a query might be prohibitive, and when an update is actually made, the magnitude of this update might be tiny.

Therefore, we set the threshold adaptively: we start s_t high, and keep dividing it by two until we reach a level where there are enough misclassifications amongst the points queried. By wrapping this filtering strategy around the modified Perceptron update, we get an active learning algorithm (Figure 4) with the following label complexity guarantee.

Theorem 3 *Pick any $\delta, \varepsilon > 0$. Consider a stream of data points x_t drawn uniformly at random from the surface of the unit sphere in $\mathbb{R}^d$, and corresponding labels y_t that are consistent with some linear*

separator. With probability $1-\delta$*, if the active modified Perceptron algorithm (Figure 4) is given a stream of* $\tilde{O}(\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$ *such unlabeled points, it will request* $\tilde{O}(d\log\frac{1}{\varepsilon})$ *labels, make* $\tilde{O}(d\log\frac{1}{\varepsilon})$ *errors (on all points, labeled or not), and have final error* $\leq \varepsilon$*.*

The proofs of Theorems 1 through 3 are in Sections 4 through 6, respectively.

2. Related Work

Much of the early theory work on active learning was in the *query learning* model, in which the learner has the ability to synthesize arbitrary data points and request their labels. See Angluin (2001) for an excellent survey of this area. In this paper, we consider a different setting, in which (1) there is an underlying joint distribution over data points and labels, (2) the learner has access to (unlabeled) data points drawn at random from this distribution, and (3) the learner is able to request labels only for points obtained in this way, not for arbitrary points. This framework for active learning was originally introduced by Cohn, Atlas, and Ladner (1994),[3] along with a simple and elegant querying algorithm. Unless we specify otherwise, we will use the term selective sampling to denote the framework. In this work, we focus on the *realizable* setting: the hypothesis class which we consider for learning contains a classifier with zero error on the data distribution.[4] Our contribution to active learning is for online learning of linear separators through the origin, under the uniform distribution.

Several methods for learning linear separators (or their probabilistic analogues) in the selective sampling framework, have been proposed in the literature. Some have been shown to work reasonably well in practice, for example Lewis and Gale's sequential algorithm for text classification (Lewis and Gale, 1994), which has batch access to the remaining unlabeled data points at each iteration. Several of these are similar in spirit to our approach, in that they query points with small margins, such as Tong and Koller's active learning algorithms that use a support vector machine (SVM) as the underlying classifier (Tong and Koller, 2001).

On the theoretical side, there have been some encouraging upper bounds on label complexity; however, some of the schemes achieving them have not yet been proven efficient. Dasgupta (2005) provided a result for learning general hypothesis classes, in a non-Bayesian, realizable setting. For homogeneous half-spaces with data distributed uniformly on the sphere, this result implies an upper bound on label complexity of $\tilde{O}(d\log^2(\frac{1}{\varepsilon}))$. Balcan, Beygelzimer, and Langford (2006) provided a technique for learning general hypothesis classes, in a non-Bayesian, agnostic setting, for which they showed a label complexity upper bound of $\tilde{O}(d^2\log\frac{1}{\varepsilon})$ for learning linear separators under the uniform input distribution.[5] Both of these results rely on schemes that are computationally prohibitive, requiring exponential storage and/or computation.

The literature contains several active learning algorithms that are both feasible to implement (at least in special cases) and have label complexity guarantees, although none of them is quite as simple as the algorithm we present in this paper. We have already discussed the label complexity upper bound attained by Freund et al. (1997) for the Query By Committee algorithm of Seung et al. (1992). More recently, it was shown how to efficiently implement this scheme for linear separators under certain prior distributions, and the empirical results were encouraging (Gilad-Bachrach et al.,

3. The conference version dates back to NIPS 1989, with a superset of the coauthors.
4. The *agnostic* setting removes this assumption.
5. This bound was further tightened to $\tilde{O}(d^{1.5}\log\frac{1}{\varepsilon})$, in Balcan et al. (2007), and this scheme has also been analyzed by Hanneke (2007), who introduced a new label complexity measure.

2005). Cesa-Bianchi et al. (2003) provided regret bounds on a selective sampling algorithm for learning linear thresholds from a stream of iid examples corrupted by random class noise whose rate scales with the examples' margins. For half-spaces under the uniform input distribution, in the realizable setting, the algorithm of Balcan et al. (2006) can be implementated efficiently, as shown by Balcan et al. (2007), which analyzed various margin-based techniques for active learning, matching our label complexity bound in the same setting. Dasgupta, Hsu, and Monteleoni (2007) recently gave an active learning algorithm for general concept classes in the non-Bayesian, agnostic setting (a generalization of the original selective sampling algorithm of Cohn et al. 1994) which, for half-spaces under the uniform input distribution, in the realizable case, has a label complexity upper bound of $\tilde{O}(d^{1.5}\log\frac{1}{\varepsilon})$.

Cesa-Bianchi et al. (2004) analyzed an algorithm which conforms to roughly the same template as ours but differs in both the update and filtering rule—it uses the regular perceptron update and it queries points x_t according to a fixed, randomized rule which favors small $|v_t \cdot x_t|$. The authors make no distributional assumptions on the input and they show that in terms of worst-case hinge-loss bounds, their algorithm does about as well as one which queries *all* labels. The actual fraction of points queried varies from data set to data set. In contrast, our objective is to achieve a target generalization error with minimum label complexity, although we also obtain a mistake bound (on both labeled and unlabeled points) under our distributional assumption.

It is known that active learning does not always give a large improvement in the sample complexity of learning linear separators. For instance, in our setting, in which data is distributed uniformly over the unit sphere, Dasgupta (2004) showed that if the target linear separator is allowed to be non-homogeneous, then the number of labels required to reach error ε is $\Omega(\frac{1}{\varepsilon})$, no matter what active learning scheme is used. This lower bound also applies to learning homogeneous linear separators with respect to an arbitrary distribution. In the fully agnostic setting, Kääriäinen (2006) provided a lower bound of $\Omega(\frac{\eta^2}{\varepsilon^2})$, where η is the error rate of the best hypothesis in the concept class.

3. Preliminaries

In our model, all data x_t lie on the surface of the unit ball in $\mathbb{R}^d$, which we denote by S:

$$S = \left\{ x \in \mathbb{R}^d \;\middle|\; \|x\| = 1 \right\}.$$

Their labels y_t are either -1 or $+1$, and the target function is a half-space $u \cdot x \geq 0$ represented by a unit vector $u \in \mathbb{R}^d$ which classifies all points perfectly, that is, $y_t(u \cdot x_t) > 0$ for all t, with probability one.

For any vector $v \in \mathbb{R}^d$, we define $\hat{v} = \frac{v}{\|v\|}$ to be the corresponding unit vector.

Our lower bound (Theorem 1) is distribution-free; thereafter we will assume that the data points x_t are drawn independently from the uniform distribution over S.

Under the uniform input distribution, any hypothesis $v \in \mathbb{R}^d$ has error

$$\varepsilon(v) = P_{x \in S}[\mathtt{SGN}(v \cdot x) \neq \mathtt{SGN}(u \cdot x)] = \frac{\arccos(u \cdot \hat{v})}{\pi}.$$

We will refer to the error rate of a hypothesis v as its *generalization error*, since in the realizable case the target itself has zero error.

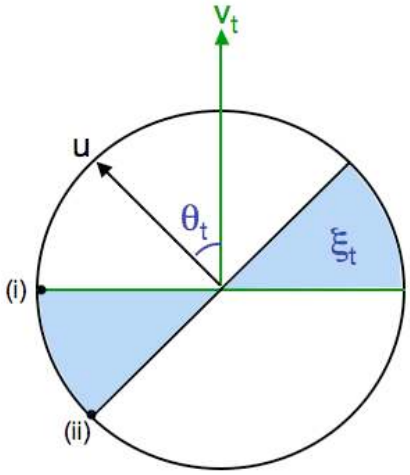

Figure 1: The projection of the error region ξ_t onto the plane defined by u and v_t.

For a hypothesis v_t, we will denote the angle between u and v_t by θ_t, and we will define the error region of v_t as $\xi_t = \{x \in S \,|\, \mathtt{SGN}(v_t \cdot x) \neq \mathtt{SGN}(u \cdot x)\}$. Figure 1 provides a schematic of the projection of the error region onto the plane defined by u and v_t.

We will use the term *margin*, in the context of learning half-spaces, to denote simply the distance from an example to the separator in question, as opposed to the standard use of this term (as the minimum over examples of this distance with respect to the target separator). For example, we will denote the margin of x with respect to v as $|x \cdot v|$.

We will use a few useful inequalities for θ on the interval $(0, \frac{\pi}{2}]$.

$$\frac{4}{\pi^2} \leq \frac{1-\cos\theta}{\theta^2} \leq \frac{1}{2}, \tag{1}$$

$$\frac{2}{\pi}\theta \leq \sin\theta \leq \theta. \tag{2}$$

Equation (1) can be verified by checking that for θ in this interval, $\frac{1-\cos\theta}{\theta^2}$ is a decreasing function, and evaluating it at the endpoints.

We will also make use of the following lemma.

Lemma 4 *For any fixed unit vector a and any $\gamma \leq 1$,*

$$\frac{\gamma}{4} \leq P_{x \in S}\left[|a \cdot x| \leq \frac{\gamma}{\sqrt{d}}\right] \leq \gamma.$$

The proof is deferred to the appendix.

4. A Lower Bound for the Perceptron Update

Consider an algorithm of the following form:

```
Pick some v_0 ∈ R^d.
Repeat for t = 0,1,2,...:
  Get some (x,y) for which y(v_t · x) ≤ 0.
  v_{t+1} = v_t + yx
```

On any update,

$$v_{t+1} \cdot u = v_t \cdot u + y(x \cdot u). \tag{3}$$

Thus, if we assume for simplicity that $v_0 \cdot u \geq 0$ (we can always just start count when this first occurs) then $v_t \cdot u \geq 0$ always, and θ_t, the angle between u and v_t is always acute. Since $\|u\| = 1$, the following holds:

$$\|v_t\| \cos\theta_t = v_t \cdot u.$$

The update rule also implies

$$\|v_{t+1}\|^2 = \|v_t\|^2 + 1 + 2y(v_t \cdot x). \tag{4}$$

Thus $\|v_t\|^2 \leq t + \|v_0\|^2$ for all t. In particular, this means that Theorem 1 is an immediate consequence of the following lemma.

Lemma 5 *Assume $v_0 \cdot u \geq 0$ (i.e., start count when this first occurs). Then*

$$\theta_{t+1} \leq \theta_t \;\Rightarrow\; \sin\theta_t \geq \min\left\{\frac{1}{3}, \frac{1}{5\|v_t\|}\right\}.$$

Proof Figure 1 shows the unit circle in the plane defined by u and v_t. The dot product of any point $x \in \mathbb{R}^d$ with either u or v_t depends only upon the projection of x into this plane. The point is misclassified when its projection lies in the shaded region. For such points, $y(u \cdot x)$ is at most $\sin\theta_t$ (point (i)) and $y(v_t \cdot x)$ is at least $-\|v_t\|\sin\theta_t$ (point (ii)).

Combining this with Equations (3) and (4), we get

$$\begin{aligned} v_{t+1} \cdot u &\leq v_t \cdot u + \sin\theta_t, \\ \|v_{t+1}\|^2 &\geq \|v_t\|^2 + 1 - 2\|v_t\|\sin\theta_t. \end{aligned}$$

To establish the lemma, we first assume $\theta_{t+1} \leq \theta_t$ and $\sin\theta_t \leq \frac{1}{5\|v_t\|}$, and then conclude that $\sin\theta_t \geq \frac{1}{3}$.

$\theta_{t+1} \leq \theta_t$ implies

$$\cos^2\theta_t \;\leq\; \cos^2\theta_{t+1} \;=\; \frac{(u \cdot v_{t+1})^2}{\|v_{t+1}\|^2} \;\leq\; \frac{(u \cdot v_t + \sin\theta_t)^2}{\|v_t\|^2 + 1 - 2\|v_t\|\sin\theta_t}.$$

The final denominator is positive since $\sin\theta_t \leq \frac{1}{5\|v_t\|}$. Rearranging,

$$(\|v_t\|^2 + 1 - 2\|v_t\|\sin\theta_t)\cos^2\theta_t \;\leq\; (u \cdot v_t)^2 + \sin^2\theta_t + 2(u \cdot v_t)\sin\theta_t,$$

and using $\|v_t\|\cos\theta_t = (u \cdot v_t)$:

$$(1 - 2\|v_t\|\sin\theta_t)\cos^2\theta_t \;\leq\; \sin^2\theta_t + 2\|v_t\|\sin\theta_t\cos\theta_t.$$

```
Inputs:  dimensionality d and budget on number of updates
(mistakes) M.
  Let v_1 = x_1 y_1 for the first example (x_1, y_1).
  For t = 1 to M:
    Let (x_t, y_t) be the next example with y(x · v_t) < 0.
    v_{t+1} = v_t − 2(v_t · x_t)x_t
```

Figure 2: The (non-active) modified Perceptron algorithm. The standard Perceptron update, $v_{t+1} = v_t + y_t x_t$, is in the same direction (note $y_t = -\mathtt{SGN}(v_t \cdot x_t)$) but different magnitude (scaled by a factor of $2|v_t \cdot x_t|$).

Again, since $\sin\theta_t \leq \frac{1}{5\|v_t\|}$, it follows that $(1-2\|v_t\|\sin\theta_t) \geq \frac{3}{5}$ and that $2\|v_t\|\sin\theta_t\cos\theta_t \leq \frac{2}{5}$. Using $\cos^2 = 1-\sin^2$, we then get

$$\frac{3}{5}(1-\sin^2\theta_t) \;\leq\; \sin^2\theta_t + \frac{2}{5},$$

which works out to $\sin^2\theta_t \geq \frac{1}{8}$, implying $\sin\theta_t > \frac{1}{3}$. ■

The problem is that the perceptron update can be too large. In $\mathbb{R}^2$ (e.g., Figure 1), when θ_t is tiny, the update will cause v_{t+1} to overshoot the mark and swing too far to the other side of u, unless $\|v_t\|$ is very large: to be precise, we need $\|v_t\| = \Omega(\frac{1}{\sin\theta_t})$. But $\|v_t\|$ grows slowly, at best at a rate of $\sqrt{t}$. If $\sin\theta_t$ is proportional to the error of v_t, as in the case of data distributed uniformly over the unit sphere, this means that the perceptron update cannot stably maintain an error rate $\leq \varepsilon$ until $t = \Omega(\frac{1}{\varepsilon^2})$.

5. The Modified Perceptron Update

We now describe a modified Perceptron algorithm. Unlike the standard Perceptron, it ensures that $v_t \cdot u$ is increasing, that is, the error of v_t is monotonically decreasing. Another difference from the standard update (and other versions) is that the magnitude of the current hypothesis, $\|v_t\|$, is always 1, which is convenient for the analysis.

The modified Perceptron algorithm is shown in Figure 2. We now show that the norm of v_t stays at one. Note that $\|v_1\| = 1$ and

$$\|v_{t+1}\|^2 = \|v_t\|^2 + 4(v_t \cdot x_t)^2\|x_t\|^2 - 4(v_t \cdot x_t)^2 = 1$$

by induction. In contrast, for the standard perceptron update, the magnitude of v_t increases steadily.

With the modified update, the error can only decrease, because $v_t \cdot u$ only increases:

$$v_{t+1} \cdot u = v_t \cdot u - 2(v_t \cdot x_t)(x_t \cdot u) = v_t \cdot u + 2|v_t \cdot x_t||x_t \cdot u|. \tag{5}$$

The second equality follows from the fact that v_t misclassified x_t. Thus $v_t \cdot u$ is increasing, and the increase can be bounded from below by showing that $|v_t \cdot x_t||x_t \cdot u|$ is large. This is a different approach from previous analyses.

Hampson and Kibler (1999) previously used this update for learning linear separators, calling it the "Reflection" method, based on the "Reflexion" method due to Motzkin and Schoenberg (1954). These names are likely due to the following geometric property of this update:

$$x_t \cdot v_{t+1} = x_t \cdot v_t - 2(x_t \cdot v_t)(x_t \cdot x_t) = -(x_t \cdot v_t).$$

In general, one can consider modified updates of the form $v_{t+1} = v_t - \alpha(v_t \cdot x_t)x_t$, which corresponds to the "Relaxation" method of solving linear inequalities (Agmon, 1954; Motzkin and Schoenberg, 1954). When $\alpha \neq 2$, the vectors v_t no longer remain of fixed length; however, one can verify that their corresponding unit vectors $\hat{v}_t$ satisfy

$$\hat{v}_{t+1} \cdot u = (\hat{v}_t \cdot u + \alpha|\hat{v}_t \cdot x_t||x_t \cdot u|)/\sqrt{1-\alpha(2-\alpha)(\hat{v}_t \cdot x_t)^2},$$

and thus any choice of $\alpha \in [0,2]$ guarantees non-increasing error. Blum et al. (1996) used $\alpha = 1$ to guarantee progress in the denominator (their analysis did not rely on progress in the numerator) as long as $\hat{v}_t \cdot u$ and $(\hat{v}_t \cdot x_t)^2$ were bounded away from 0. Their approach was used in a batch setting as one piece of a more complex algorithm for noise-tolerant learning. In our sequential framework, we can bound $|\hat{v}_t \cdot x_t||x_t \cdot u|$ away from 0 in expectation, under the uniform distribution, and hence the choice of $\alpha = 2$ is most convenient, but $\alpha = 1$ would work as well. Although we do not further optimize our choice of the constant α, this choice itself may yield interesting future work, perhaps by allowing it to be a function of the dimension.

5.1 Analysis of (Non-Active) Modified Perceptron

How large do we expect $|v_t \cdot x_t|$ and $|u \cdot x_t|$ to be for an error (x_t, y_t)? As we shall see, in d dimensions, one expects each of these terms to be on the order of $d^{-1/2}\sin\theta_t$, where $\sin\theta_t = \sqrt{1-(v_t \cdot u)^2}$. Hence, we might expect their product to be about $(1-(v_t \cdot u)^2)/d$, which is how we prove the following lemma.

Note, we have made little effort to optimize constant factors.

Lemma 6 *For any v_t, with probability at least $\frac{1}{3}$,*

$$1 - v_{t+1} \cdot u \leq (1 - v_t \cdot u)\left(1 - \frac{1}{50d}\right).$$

There exists a constant $c > 0$, such that with probability at least $\frac{63}{64}$, for any v_t,

$$1 - v_{t+1} \cdot u \leq (1 - v_t \cdot u)\left(1 - \frac{c}{d}\right).$$

Proof We show only the first part of the lemma. The second part is quite similar. We will argue that each of $|v_t \cdot x_t|, |u \cdot x_t|$ is "small" with probability at most 1/3. This means, by the union bound, that with probability at least 1/3, they are both sufficiently large.

The error rate of v_t is θ_t/π, where $\cos\theta_t = v_t \cdot u$. Also define the error region $\xi_t = \{x \in S \,|\, \mathtt{SGN}(v_t \cdot x) \neq \mathtt{SGN}(u \cdot x)\}$. By Lemma 4, for an x drawn uniformly from the sphere,

$$P_{x \in S}\left[|v_t \cdot x| \leq \frac{\theta_t}{3\pi\sqrt{d}}\right] \leq \frac{\theta_t}{3\pi}.$$

Using $P[A|B] \leq P[A]/P[B]$, we have,

$$P_{x\in S}\left[|v_t \cdot x| \leq \frac{\theta_t}{3\pi\sqrt{d}} \;\middle|\; x \in \xi_t \right] \leq \frac{P_{x\in S}[|v_t \cdot x| \leq \frac{\theta_t}{3\pi\sqrt{d}}]}{P_{x\in S}[x \in \xi_t]} \leq \frac{\theta_t/(3\pi)}{\theta_t/\pi} = \frac{1}{3}.$$

Similarly for $|u \cdot x|$, and by the union bound the probability that $x \in \xi_t$ is within margin $\frac{\theta}{3\pi\sqrt{d}}$ from either u or v is at most $\frac{2}{3}$. Since the updates only occur if x is in the error region, we now have a lower bound on the expected magnitude of $|v_t \cdot x||u \cdot x|$:

$$P_{x\in S}\left[|v_t \cdot x||u \cdot x| \geq \frac{\theta_t^2}{(3\pi\sqrt{d})^2} \middle| x \in \xi_t \right] \geq \frac{1}{3}.$$

Hence, we know that with probability at least $1/3$, $|v_t \cdot x||u \cdot x| \geq \frac{1-(v_t\cdot u)^2}{100d}$, since $\theta_t^2 \geq \sin^2\theta_t = 1-(v_t \cdot u)^2$ and $(3\pi)^2 < 100$. In this case,

$$\begin{aligned} 1 - v_{t+1}\cdot u &\leq 1 - v_t \cdot u - 2|v_t \cdot x_t||u \cdot x_t| \\ &\leq 1 - v_t \cdot u - \frac{1-(v_t\cdot u)^2}{50d} \\ &= (1 - v_t \cdot u)\left(1 - \frac{1+v_t\cdot u}{50d}\right), \end{aligned}$$

where the first inequality is by application of (5). ■

Finally, we give a high-probability bound, that is, Theorem 2, stated here with proof.

Theorem 7 *With probability $1-\delta$ with respect to the uniform distribution on the unit sphere, in the supervised, realizable setting, after $M = O(d(\log\frac{1}{\varepsilon} + \log\frac{1}{\delta}))$ mistakes, the generalization error of the modified Perceptron algorithm is at most ε.*

Proof By the above lemma, we can conclude that, for any vector v_t,

$$E[1 - v_{t+1}\cdot u] \leq (1 - v_t \cdot u)\left(1 - \frac{1}{3(50d)}\right).$$

This is because with $\geq 1/3$ probability it goes down by a factor of $1 - \frac{1}{50d}$ and with the remaining $\leq 2/3$ probability it does not increase. Hence, after M mistakes,

$$E[1 - v_M \cdot u] \leq (1 - v_1 \cdot u)\left(1 - \frac{1}{150d}\right)^M \leq \left(1 - \frac{1}{150d}\right)^M,$$

since $v_1 \cdot u \geq 0$. By Markov's inequality,

$$P\left[1 - v_M \cdot u \geq \left(1 - \frac{1}{150d}\right)^M \delta^{-1}\right] \leq \delta.$$

Finally, using (1) and $\cos\theta_M = v_M \cdot u$, we see $P[\frac{4}{\pi^2}\theta_M^2 \geq (1-\frac{1}{150d})^M\delta^{-1}] \leq \delta$. Using $M = 150d\log\frac{1}{\varepsilon\delta}$ gives $P[\frac{\theta_M}{\pi} \geq \varepsilon] \leq \delta$, as required. ■

The additional factor of $\frac{1}{\varepsilon}$ in the bound on unlabeled samples ($\tilde{O}(\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$) follows by upper bounding the number of unlabeled samples until an update: when the hypothesis has error rate ε, the waiting time (in samples) until an update is $\frac{1}{\varepsilon}$, in expectation.

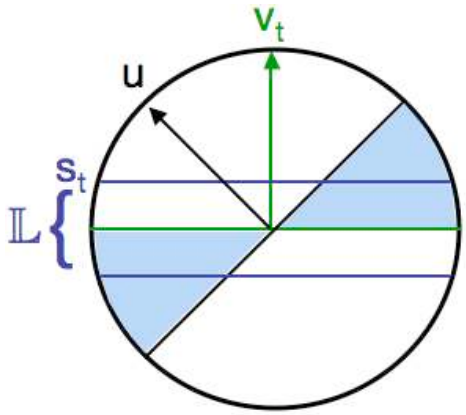

Figure 3: The active learning rule is to query for labels on points x in $\mathbb{L}$ which is defined by the threshold s_t on $|v_t \cdot x|$.

6. An Active Modified Perceptron

The ideal objective in designing an active learning rule that minimizes label complexity would be to query for labels only on points in the error region, ξ_t. However without knowledge of u, the algorithm is unaware of the location of ξ_t. The intuition behind our active learning rule is to approximate the error region, given the information the algorithm does have: v_t. As shown in Figure 3, the labeling region $\mathbb{L}$ is simply formed by thresholding the margin of a candidate example with respect to v_t.

The active version of the modified Perceptron algorithm is shown in Figure 4. The algorithm is similar to the algorithm of the previous section, in its update step. For its filtering rule, we maintain a threshold s_t and we only ask for labels of examples with $|v_t \cdot x| \leq s_t$. Approximating the error region is achieved by choosing the threshold, s_t, adaptively, so as to manage the tradeoff between $\mathbb{L}$ being too large, causing many labels to be wasted without hitting ξ_t (and thus yielding updates), and $\mathbb{L}$ only containing points with very small margins with respect to v_t, since our update step will make very small updates on such points. We decrease the threshold adaptively over time, starting at $s_1 = 1/\sqrt{d}$ and reducing it by a factor of two whenever we have a run of labeled examples on which we are correct.

For Theorem 3, we select values of R, L that yield ε error with probability at least $1-\delta$. The idea of the analysis is as follows:

Definition 7 *We say the tth update is "good" if,*

$$1 - v_{t+1} \cdot u \leq (1 - v_t \cdot u)\left(1 - \frac{c}{d}\right).$$

(The constant c is from Lemma 6.)

1. (Lemma 8) First, we argue that s_t is not too small (we do not decrease s_t too quickly). Assuming this is the case, then 2 and 3 hold.

2. (Lemma 10) We query for labels on at least an expected 1/32 of all *errors*. In other words, some errors may go undetected because we do not ask for their labels, but the number of

Inputs: Dimensionality d, maximum number of labels L,
and patience R.
  $v_1 = x_1 y_1$ for the first example (x_1, y_1).
  $s_1 = 1/\sqrt{d}$
  For $t = 1$ **to** L:
    Wait for the next example x : $|x \cdot v_t| \leq s_t$ and query its label.
    Call this labeled example (x_t, y_t).
    If $(x_t \cdot v_t) y_t < 0$, **then**:
      $v_{t+1} = v_t - 2(v_t \cdot x_t) x_t$
      $s_{t+1} = s_t$
    else:
      $v_{t+1} = v_t$
      If predictions were correct on R consecutive labeled
      examples (i.e., $(x_i \cdot v_i) y_i \geq 0 \ \forall i \in \{t-R+1, t-R+2, \ldots, t\}$),
      then set $s_{t+1} = s_t/2$, **else** $s_{t+1} = s_t$.

Figure 4: An active version of the modified Perceptron algorithm.

mistakes total should not be much more than 32 times the number of updates we actually perform.

3. (Lemma 11) Each update is *good* (Definition 7) with probability at least $1/2$.

4. (Theorem 3) Finally, we conclude that we cannot have too many label queries, updates, or total errors, because half of our updates are good, 1/32 of our errors are updates, and about $1/R$ of our labels are updates.

We first lower-bound s_t with respect to our error, showing that, with high probability, the threshold s_t is never too small.

Lemma 8 *With probability at least* $1 - L\left(\frac{3}{4}\right)^R$, *we have:*

$$s_t \geq \sqrt{\frac{1-(u \cdot v_t)^2}{16d}} \text{ for } t = 1, 2, \ldots, L, \text{ simultaneously.} \tag{6}$$

Before proving this lemma, it will be helpful to show the following lemma. As before, let us define $\xi_t = \{x \in S | (x \cdot v_t)(x \cdot u) < 0\}$.

Lemma 9 *For any* $\gamma \in \left(0, \sqrt{\frac{1-(u \cdot v_t)^2}{4d}}\right]$,

$$P_{x_t \in S}\left[x_t \in \xi_t \;\middle|\; |x_t \cdot v_t| < \gamma\right] \geq \frac{1}{4}.$$

Proof Let x be a random example from S such that $|x \cdot v_t| < \gamma$ and, without loss of generality, suppose that $0 \leq x \cdot v_t \leq \gamma$. Then we want to calculate the probability we err, that is, $u \cdot x < 0$. We

can decompose $x = x' + (x \cdot v_t)v_t$ where $x' = x - (x \cdot v_t)v_t$ is the component of x orthogonal to v_t, that is, $x' \cdot v_t = 0$. Similarly for $u' = u - (u \cdot v_t)v_t$. Hence,

$$u \cdot x = (u' + (u \cdot v_t)v_t) \cdot (x' + (x \cdot v_t)v_t) = u' \cdot x' + (u \cdot v_t)(x \cdot v_t).$$

In other words, we err iff $u' \cdot x' < -(u \cdot v_t)(x \cdot v_t)$. Using $u \cdot v_t \in [0,1]$ and since $x \cdot v_t \in [0, \sqrt{(1-(u \cdot v_t)^2)/(4d)}]$, we conclude that if

$$u' \cdot x' < -\sqrt{\frac{1-(u \cdot v_t)^2}{4d}}, \tag{7}$$

then we must err. Also, let $\hat{x}' = \frac{x'}{\|x'\|}$ be the unit vector in the direction of x'. It is straightforward to check that $\|x'\| = \sqrt{1-(x \cdot v_t)^2}$. Similarly, for u we define $\hat{u}' = \frac{u'}{\sqrt{1-(u \cdot v_t)^2}}$. Substituting these into (7), we must err if, $\hat{u}' \cdot \hat{x}' < -1/\sqrt{4d(1-(x \cdot v_t))^2}$, and since $\sqrt{1-(x \cdot v_t)^2} \geq \sqrt{1-1/(4d)}$, it suffices to show that,

$$P_{x \in S}\left[\hat{u}' \cdot \hat{x}' < \frac{-1}{\sqrt{4d(1-1/(4d))}} \;\middle|\; 0 \leq x \cdot v_t \leq \gamma\right] \geq \frac{1}{4}.$$

What is the probability that this happens? Well, one way to pick $x \in S$ would be to first pick $x \cdot v_t$ and then to pick $\hat{x}'$ uniformly at random from the set $S' = \{\hat{x}' \in S | \hat{x}' \cdot v_t = 0\}$, which is a unit sphere in one fewer dimensions. Hence the above probability does not depend on the conditioning. By Lemma 4, for any unit vector $a \in S'$, the probability that $|\hat{u}' \cdot a| \leq 1/\sqrt{4(d-1)}$ is at most $1/2$, so with probability at least $1/4$ (since the distribution is symmetric), the signed quantity $\hat{u}' \cdot \hat{x}' < -1/\sqrt{4(d-1)} < -1/\sqrt{4d(1-1/(4d))}$. ■

We are now ready to prove Lemma 8.

Proof [of Lemma 8] Suppose that condition (6) fails to hold for some t's. Let t be the smallest number such that (6) fails. By our choice of s_1, clearly $t > 1$. Moreover, since t is the smallest such number, and $u \cdot v_t$ is increasing, it must be the case that $s_t = s_{t-1}/2$, that is we just saw a run of R labeled examples (x_i, y_i), for $i = t-R, \ldots, t-1$, with no mistakes, $v_i = v_t$, and

$$s_i = 2s_t < \sqrt{\frac{1-(u \cdot v_t)^2}{4d}} = \sqrt{\frac{1-(u \cdot v_i)^2}{4d}}. \tag{8}$$

Such an event is highly unlikely, however, for any t. In particular, from Lemma 9, we know that the probability of (8) holding for any particular i and the algorithm not erring is at most $3/4$. Thus the chance of having any such run of length R is at most $L(3/4)^R$. ■

Lemma 9 also tells us something interesting about the fraction of errors that we are missing because we do not ask for labels. In particular,

Lemma 10 *Given that $s_t \geq \sqrt{(1-(u \cdot v_t)^2)/(16d)}$, upon the tth update, each erroneous example is queried with probability at least 1/32, that is,*

$$P_{x \in S}\left[|x \cdot v_t| \leq s_t \;\middle|\; x \in \xi_t\right] \geq \frac{1}{32}.$$

Proof Using Lemmas 9 and 4, we have

$$\begin{aligned} P_{x\in S}\left[x\in\xi_t \wedge |x\cdot v_t|\le s_t\right] &\ge P_{x\in S}\left[x\in\xi_t \wedge |x\cdot v_t|\le \sqrt{\frac{1-(u\cdot v_t)^2}{16d}}\right] \\ &\ge \frac{1}{4}P_{x\in S}\left[|x\cdot v_t|\le \sqrt{\frac{1-(u\cdot v_t)^2}{16d}}\right] \\ &\ge \frac{1}{64}\sqrt{1-(u\cdot v_t)^2} = \frac{1}{64}\sin\theta_t \\ &\ge \frac{\theta_t}{32\pi}. \end{aligned}$$

For the last inequality, we have used (2). However, $P_{x\in S}[x\in\xi_t]=\theta_t/\pi$, so we are querying an error $x\in\xi_t$ with probability at least $1/32$, that is, the above inequality implies,

$$P_{x\in S}\left[|x\cdot v_t|\le s_t \;\middle|\; x\in\xi_t\right] = \frac{P_{x\in S}\left[x\in\xi_t\wedge|x\cdot v_t|\le s_t\right]}{P_{x\in S}[x\in\xi_t]} \ge \frac{\theta_t/(32\pi)}{\theta_t/\pi} = \frac{1}{32}.$$

■

Next, we show that the updates are likely to make progress.

Lemma 11 *Assuming that $s_t \ge \sqrt{(1-(u\cdot v_t)^2)/(16d)}$, a random update is good with probability at least $1/2$, that is,*

$$P_{x_t\in S}\left[(1-v_{t+1}\cdot u)\le(1-v_t\cdot u)\left(1-\frac{c}{d}\right)\;\middle|\;|x\cdot v_t|\le s_t\wedge x_t\in\xi_t\right]\ge\frac{1}{2}.$$

Proof By Lemma 10, each error is queried with probability 1/32. On the other hand, by Lemma 6 of the previous section, 63/64 of all errors are good. Since we are querying at least 2/64 fraction of all errors, at least half of our queried errors must be good. ■

We now have the pieces to guarantee the convergence rate of the active algorithm, thereby proving Theorem 3. This involves bounding both the number of labels that we query as well as the number of total errors, which includes updates as well as errors that were never detected.

Theorem 3 *With probability $1-\delta$ with respect to the uniform distribution on the unit sphere, in the realizable setting, using $L=O\big(d\log\big(\frac{1}{\varepsilon\delta}\big)(\log\frac{d}{\delta}+\log\log\frac{1}{\varepsilon})\big)$ labels and making a total number of errors of $O\big(d\log\big(\frac{1}{\varepsilon\delta}\big)(\log\frac{d}{\delta}+\log\log\frac{1}{\varepsilon})\big)$, the final error of the active modified Perceptron algorithm will be ε, when run with the above L and $R=O(\log\frac{d}{\delta}+\log\log\frac{1}{\varepsilon})$.*

Proof Let U be the number of updates performed. We know, by Lemma 8 that with probability $1-L(\frac{3}{4})^R$,

$$s_t \ge \frac{\sin\theta_t}{4\sqrt{d}} \ge \frac{\theta_t}{2\pi\sqrt{d}} \tag{9}$$

for all t. Again, we have used (2). By Lemma 11, we know that for each t which is an update, either (9) fails or

$$E[1-u\cdot v_{t+1}|v_t]\le(1-u\cdot v_t)\left(1-\frac{c}{2d}\right).$$

Hence, after U updates, using Markov's inequality,

$$P\left[1-u\cdot v_L \geq \frac{4}{\delta}\left(1-\frac{c}{2d}\right)^U\right] \leq \frac{\delta}{4}+L\left(\frac{3}{4}\right)^R.$$

In other words, with probability $1-\frac{\delta}{4}-L(\frac{3}{4})^R$, we also have

$$U \leq \frac{2d}{c}\log\frac{4}{\delta(1-u\cdot v_L)} \leq \frac{2d}{c}\log\frac{\pi^2}{\delta\theta_L^2} = O\left(d\log\frac{1}{\delta\varepsilon}\right),$$

where for the last inequality we used (1). In total, $L \leq R\left(U+\log_2\frac{1}{s_L}\right)$. This is because once every R labels we either have at least one update or we decrease s_L by a factor of 2. Equivalently, $s_L \leq 2^{U-L/R}$. Hence, with probability $1-\frac{\delta}{4}-L(\frac{3}{4})^R$,

$$\frac{\theta_L}{2\pi\sqrt{d}} \leq s_L \leq 2^{O(d\log\frac{1}{\delta\varepsilon})-L/R}.$$

Working backwards, we choose $L/R=\Theta(d\log\frac{1}{\varepsilon\delta})$ so that the above expression implies $\frac{\theta_L}{\pi}\leq\varepsilon$, as required. We choose

$$R = 10\log\frac{2L}{\delta R} = \Theta\left(\log\frac{d\log\frac{1}{\varepsilon\delta}}{\delta}\right) = O\left(\log\frac{d}{\delta}+\log\log\frac{1}{\varepsilon}\right).$$

The first equality ensures that $L(\frac{3}{4})^R\leq\frac{\delta}{4}$. Hence, for the L and R chosen in the theorem, with probability $1-\frac{3}{4}\delta$, we have error $\frac{\theta_L}{\pi}<\varepsilon$. Finally, either condition (9) fails or each error is queried with probability at least $\frac{1}{32}$. By the multiplicative Chernoff bound, if there were a total of $E>64U$ errors, with probability $\geq 1-\frac{\delta}{4}$, at least $E/64>U$ would have been caught and used as updates. Hence, with probability at most $1-\delta$, we have achieved the target error using the specified number of labels and observing the specified number of errors. ■

7. Discussion and Conclusions

In the evolving theory of active learning, the most concrete, nontrivial scenario in which active learning has been shown to give an exponential improvement in sample complexity is that of learning a linear separator for data distributed uniformly over the unit sphere. In this paper, we have demonstrated that this particular case can be solved by a much simpler algorithm than was previously known. Table 1 summarizes our contributions in context. We report bounds for all the algorithms with respect to our setting: learning homogeneous half-spaces when the data distribution is uniform on the unit sphere and separable through the origin, although a few of the algorithms were designed for more general distributions. This paper gives the lower bounds stated for Perceptron, and provides an algorithm that attains the upper bounds in the bottom row. While we list a host of results for comparison, it is important to note that the algorithm of Dasgupta (2005) has not shown to be efficiently implementable, and that we state bounds for this realizable problem, even though the algorithm of Balcan et al. (2007) can handle certain types of noise, and A^2 (Balcan et al., 2006) and the algorithm of Dasgupta et al. (2007) were designed to handle the agnostic setting.

	Samples	Mistakes	Labels	Noise tolerance
PAC bounds	$\tilde{O}(\frac{d}{\varepsilon})$, $\Omega(\frac{d}{\varepsilon})$			
Perceptron	$\tilde{O}(\frac{d}{\varepsilon^3})$, $\Omega(\frac{1}{\varepsilon^2})$	$\tilde{O}(\frac{d}{\varepsilon^2})$, $\Omega(\frac{1}{\varepsilon^2})$	$\Omega(\frac{1}{\varepsilon^2})$	Unknown
D'05	$\tilde{O}(\frac{d}{\varepsilon}\log^2\frac{1}{\varepsilon})$	$\tilde{O}(d\log^2\frac{1}{\varepsilon})$	$\tilde{O}(d\log^2\frac{1}{\varepsilon})$	No
A^2	$\tilde{O}(\frac{d^{1.5}}{\varepsilon}\log\frac{1}{\varepsilon})$	$\tilde{O}(d^{1.5}\log\frac{1}{\varepsilon})$	$\tilde{O}(d^{1.5}\log\frac{1}{\varepsilon})$	Yes
DHM'07	$\tilde{O}(\frac{d^{1.5}}{\varepsilon}\log\frac{1}{\varepsilon})$	$\tilde{O}(d^{1.5}\log\frac{1}{\varepsilon})$	$\tilde{O}(d^{1.5}\log\frac{1}{\varepsilon})$	Yes
QBC	$\tilde{O}(\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$	$\tilde{O}(d\log\frac{1}{\varepsilon})$	$\tilde{O}(d\log\frac{1}{\varepsilon})$	No
BBZ'07	$\tilde{O}(\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$	$\tilde{O}(d\log\frac{1}{\varepsilon})$	$\tilde{O}(d\log\frac{1}{\varepsilon})$	Yes
Our algorithm	$\tilde{O}(\frac{d}{\varepsilon}\log\frac{1}{\varepsilon})$	$\tilde{O}(d\log\frac{1}{\varepsilon})$	$\tilde{O}(d\log\frac{1}{\varepsilon})$	Unknown

Table 1: Results in context, for learning half-spaces through the origin. The last column indicates whether each algorithm has been proved to exhibit some noise tolerance when used for active learning.

In all these papers, the uniform distribution of data has consistently proved amenable to analysis. This is an impressive distribution to learn against because it is difficult in some ways—most of the data is close to the decision boundary, for instance—but a more common assumption would be to make the two classes Gaussian, or to merely stipulate that they are separated by a margin. As a modest step towards relaxing this distributional assumption, we can show an (at most) polynomial dependence of the label complexity on λ (Monteleoni, 2006), when the input distribution is λ-similar to uniform, a setting studied in Freund et al. (1997).

Our algorithm is in some ways fine-tuned for linearly-separable data that are distributed uniformly; for instance, in the choice of the parameters R and s_1. An immediate open problem is therefore the following:

1. Design a version of the algorithm that is sensible for general data distributions which may not be linearly separable.

2. What types of noise can be tolerated by this scheme?

3. For what distributions can its label complexity be analyzed?

A step towards the practical realization of our algorithm is the work of Monteleoni and Kääriäinen (2007), which applies a version of it to an optical character recognition problem.

Acknowledgments

This work was done while ATK was at the Toyota Technological Institute at Chicago. Much of this work was done while CM was visiting the Toyota Technological Institute at Chicago. Some of this work was done while CM was at the Massachusetts Institute of Technology, Computer Science and Intelligence Laboratory, and at the University of California, San Diego, Department of Computer Science and Engineering. CM would like to thank Adam Klivans, Brendan McMahan, and Vikas Sindhwani, for various discussions at TTI, and David McAllester for the opportunity to visit. The authors thank the anonymous reviewers of COLT 2005 and JMLR for helpful comments used in revision.

Appendix A. Proof of Lemma 4

Proof [Lemma 4] Let $r = \gamma/\sqrt{d}$ and let A_d be the area of a d-dimensional unit sphere, that is, the surface of a $(d+1)$-dimensional unit ball. Then

$$P_x[|a \cdot x| \leq r] = \frac{\int_{-r}^{r} A_{d-2}(1-z^2)^{\frac{d-2}{2}}(1-z^2)^{-1/2}dz}{A_{d-1}} = \frac{2A_{d-2}}{A_{d-1}}\int_0^r (1-z^2)^{(d-3)/2}dz.$$

First observe,

$$r(1-r^2)^{(d-3)/2} \leq \int_0^r (1-z^2)^{(d-3)/2}dz \leq r. \tag{10}$$

For $x \in [0, 0.5]$, $1-x \geq 4^{-x}$. Hence, for $0 \leq r \leq 2^{-1/2}$,

$$(1-r^2)^{(d-3)/2} \geq 4^{-r^2((d-3)/2)} \geq 2^{-r^2 d}.$$

So we can conclude that the integral of (10) is in $[r/2, r]$ for $r \in [0, 1/\sqrt{d}]$. The ratio $2A_{d-2}/A_{d-1}$ can be shown to be in the range $[\sqrt{d/3}, \sqrt{d}]$ by straightforward induction on d, using the definition of the Γ function, and the fact that $A_{d-1} = 2\pi^{d/2}/\Gamma(d/2)$. ■

References

S. Agmon. The relaxation method for linear inequalities. *Canadian Journal of Math.*, 6(3):382–392, 1954.

D. Angluin. Queries revisited. *In Proc. 12th International Conference on Algorithmic Learning Theory*, LNAI,2225:12–31, 2001.

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proc. International Conference on Machine Learning*, 2006.

M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. In *Proc. 20th Annual Conference on Learning Theory*, 2007.

E.B. Baum. The perceptron algorithm is fast for nonmalicious distributions. *Neural Computation*, 2:248–260, 1997.

A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. In *Proc. 37th Annual IEEE Symposium on the Foundations of Computer Science*, 1996.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. Learning probabilistic linear-threshold classifiers via selective sampling. In *Proc. 16th Annual Conference on Learning Theory*, 2003.

N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear-threshold algorithms. In *Advances in Neural Information Processing Systems 17*, 2004.

D.A. Cohn, L. Atlas, and R.E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, 2004.

S. Dasgupta. Coarse sample complexity bounds for active learning. In *Advances in Neural Information Processing Systems 18*, 2005.

S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *Advances in Neural Information Processing Systems*, 2007.

Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

R. Gilad-Bachrach, A. Navot, and N. Tishby. Query by committee made real. In *Advances in Neural Information Processing Systems 18*, 2005.

S. Hampson and D. Kibler. Minimum generalization via reflection: A fast linear threshold learner. *Machine Learning*, 37(1):51–73, 1999.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *Proc. International Conference on Machine Learning*, 2007.

M. Kääriäinen. Active learning in the non-realizable case. In *Proc. 17th International Conference on Algorithmic Learning Theory*, 2006.

D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In *Proc. of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, 1994.

C. Monteleoni and M. Kääriäinen. Practical online active learning for classification. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, Online Learning for Classification Workshop*, 2007.

C.E. Monteleoni. *Learning with Online Constraints: Shifting Concepts and Active Learning*. PhD Thesis, MIT Computer Science and Artificial Intelligence Laboratory, 2006.

T.S. Motzkin and I.J. Schoenberg. The relaxation method for linear inequalities. *Canadian Journal of Math.*, 6(3):393–404, 1954.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

R.A. Servedio. On PAC learning using winnow, perceptron, and a perceptron-like algorithm. In *Computational Learning Theory*, pages 296 – 307, 1999.

H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proc. Fifth Annual ACM Conference on Computational Learning Theory*, 1992.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.