Efficient Active Learning of Halfspaces: an Aggressive Approach

Alon Gonen ALONGNN@CS.HUJI.AC.IL
Benin School of CSE, The Hebrew University

Sivan Sabato SIVAN.SABATO@MICROSOFT.COM
Microsoft Reserach New England

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
Benin School of CSE, The Hebrew University

Abstract

We study pool-based active learning of half-spaces. We revisit the aggressive approach for active learning in the realizable case, and show that it can be made efficient and practical, while also having theoretical guarantees under reasonable assumptions. We further show, both theoretically and experimentally, that it can be preferable to mellow approaches. Our efficient aggressive active learner of half-spaces has formal approximation guarantees that hold when the pool is separable with a margin. While our analysis is focused on the realizable setting, we show that a simple heuristic allows using the same algorithm successfully for pools with low error as well. We further compare the aggressive approach to the mellow approach, and prove that there are cases in which the aggressive approach results in significantly better label complexity compared to the mellow approach. Experiments demonstrate that substantial improvements in label complexity can be achieved using the aggressive approach, in realizable and low-error settings.

1. Introduction

We consider pool-based active learning (McCallum & Nigam, 1998), in which a learner receives a pool of unlabeled examples, and can iteratively query a teacher for the labels of examples from the pool. The goal of the learner is to return a low-error prediction rule for the labels of the examples, using a small number of queries. The number of queries used by the learner is termed its *label complexity*. This setting is most useful when unlabeled data is abundant but labeling is expensive, a common case in many data-laden applications. A pool-based algorithm can be used to learn a classifier in the standard PAC model, while querying fewer labels. This can be done by first drawing a random unlabeled sample to be used as the pool, then using pool-based active learning to identify its labels with few queries, and then using the resulting labeled sample as input to a regular "passive" PAC-learner.

Most active learning approaches can be loosely described as more 'aggressive' or more 'mellow'. A more aggressive approach is one in which only 'highly informative' queries are requested (Tong & Koller, 2002; Balcan et al., 2007; Dasgupta et al., 2005), while the mellow approach, first proposed in the CAL algorithm (Cohn et al., 1994), is one in which the learner essentially queries all the labels it has not inferred yet.

In recent years a significant advancement has been made for active learning in the PAC model. In particular, it has been shown that when the data is realizable (that is, incurs zero error under some assumed hypothesis class), the mellow approach can guarantee an exponential improvement in label complexity, compared to passive learning (Balcan et al., 2006). This exponential improvement depends on the properties of the distribution, as quantified by the *Disagreement Coefficient* proposed by Hanneke (2007). Specifically, when learning half-spaces in Euclidean space, the disagreement coefficient implies a low label complexity when the data distribution is uniform or close to uniform. El-Yaniv & Wiener (2012) have shown guarantees if the data distribution is a finite mixture of Gaussians.

An advantage of the mellow approach is its ability

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

to obtain label complexity improvements in the agnostic setting, which allows an arbitrary and large labeling error (Balcan et al., 2006; Dasgupta et al., 2007). Nonetheless, in the realizable case the mellow approach is not always optimal, even for the uniform distribution (Balcan et al., 2007). In this work we revisit the aggressive approach for the realizable case, and in particular for active learning of half-spaces in Euclidean space. We show that it can be made efficient and practical, while also having theoretical guarantees under reasonable assumptions. We further show, both theoretically and experimentally, that it can sometimes be preferable to mellow approaches.

In the first part of this work we construct an efficient aggressive active learner for half-spaces in Euclidean space, which is approximately optimal if the pool is separable with a margin. While our analysis is focused on the realizable setting, we show that a simple heuristic allows using the same algorithm successfully for pools with low error as well. Our algorithm for halfspaces is based on a greedy query selection approach as proposed in Tong & Koller (2002); Dasgupta (2005). We obtain improved target-dependent approximation guarantees for greedy selection in a general active learning setting. These guarantees allow us to prove meaningful approximation guarantees for halfspaces based on a margin assumption.

In the second part of this work we compare the greedy approach to the mellow approach. We prove that there are cases in which this highly aggressive greedy approach results in significantly better label complexity compared to the mellow approach. We demonstrate experimentally that substantial improvements in label complexity can be achieved compared to mellow approaches, for both realizable and low-error settings.

The first greedy query selection algorithm for learning halfspaces in Euclidean space was proposed by Tong & Koller (2002). The greedy algorithm is based on the notion of a *version space*: the set of all hypotheses in the hypothesis class that are consistent with the labels currently known to the learner. In the case of halfspaces, each version space is a convex body in Euclidean space. Each possible query thus splits the current version space into two parts: the version space that would result if the query received a positive label, and the one resulting from a negative label. Tong and Koller proposed to query the example from the pool that splits the version space as evenly as possible. To implement this policy, one would need to calculate the volume of a convex body in Euclidean space, a problem which is known to be computationally intractable (Brightwell & Winkler, 1991). Tong and Koller thus implemented several heuristics that attempt to follow their proposed selection principle using an efficient algorithm. For instance, they suggest to choose the example which is closest to the max-margin solution of the data labeled so far. However, none of their heuristics provably follow this greedy selection policy.

The label complexity of greedy pool-based active learning algorithms can be analyzed by comparing it to the best possible label complexity of any pool-based active learner on the same pool. The *worst-case label complexity* of an active learner is the maximal number of queries it would make on the given pool, where the maximum is over all the possible classification rules that can be consistent with the pool according to the given hypothesis class. The *average-case label complexity* of an active learner is the average number of queries it would make on the given pool, where the average is taken with respect to some fixed probability distribution P over the possible classifiers in the hypothesis class. For each of these definitions, the optimal label complexity is the lowest label complexity that can be achieved by an active learner on the given pool. Since implementing the optimal label complexity is usually computationally intractable, an alternative is to implement an efficient algorithm, and to guarantee a bounded factor of approximation on its label complexity, compared to the optimal label complexity.

Dasgupta (2005) showed that if a greedy algorithm splits the probability mass of the version space as evenly as possible, as defined by the fixed probability distribution P over the hypothesis class, then the approximation factor for its average label complexity, with respect to the same distribution, is bounded by $O(\log(1/p_{\min}))$, where $p_{\min}$ is the minimal probability of any possible labeling of the pool, if the classifier is drawn according to the fixed distribution. Golovin & Krause (2010) extended Dasgupta's result and showed that a similar bound holds for an approximate greedy rule. They also showed that the approximation factor for the worst-case label complexity of an approximate greedy rule is also bounded by $O(\log(1/p_{\min}))$, thus extending a result of Arkin et al. (1993). Note that in the worst-case analysis, the fixed distribution is only an analysis tool, and does not represent any assumption on the true probability of the possible labelings.

Returning to greedy selection of halfspaces in Euclidean space, we can see that the fixed distribution over hypotheses that matches the volume-splitting strategy is the distribution that draws a halfspace uniformly from the unit ball. The analysis presented above thus can result in poor approximation factors, since if there are instances in the pool that are very close to each other, then $p_{\min}$ might be very small.

We first show that mild conditions suffice to guarantee that $p_{\min}$ is bounded from below. By proving a variant of a result due to Muroga et al. (1961), we show that if the examples in the pool are stored using number of a finite accuracy $1/c$, then $p_{\min} \geq (c/d)^{d^2}$, where d is the dimensionality of the space. It follows that the approximation factor for the worst-case label complexity of our algorithm is at most $O(d^2 \log(d/c))$.

While this result provides us with a uniform lower bound on $p_{\min}$, in many real-world situations the probability of the target hypothesis (i.e., one that is consistent with the true labeling) could be much larger than $p_{\min}$. A noteworthy example is when the target hypothesis separates the pool with a margin of γ. In this case, it can be shown that the probability of the target hypothesis is at least γ^d, which can be significantly larger than $p_{\min}$. An immediate question is therefore: can we obtain a *target-dependent* label complexity approximation factor that would depend on the probability of the target hypothesis, $P(h)$, instead of the minimal probability of any labeling?

We prove that such a target dependent bound *does not* hold for a general approximate-greedy algorithm. To overcome this, we introduce an algorithmic change to the approximate greedy policy, which allows us to obtain a label complexity approximation factor of $\log(1/P(h))$. This can be achieved by running the approximate-greedy procedure, but stopping the procedure early, before reaching a pure version space that exactly matches the labeling of the pool. Then, an approximate majority vote over the version space can be used to determine the labels of the pool. This result is general and holds for any hypothesis class and distribution. For halfspaces, it implies an approximation-factor guarantee of $O(d \log(1/\gamma))$.

We use this result to provide an efficient approximately-optimal active learner for halfspaces, called *ALuMA*, which relies on randomized approximation of the volume of the version space (Kannan et al., 1997). This allows us to prove a margin-dependent approximation factor guarantee for ALuMA. The assumption of separation with a margin can be relaxed if a lower bound on the total hinge-loss of the best separator for the pool can be assumed. We show that under such an assumption a simple transformation on the data allows running ALuMA as if the data was separable with a margin. This results in approximately optimal label complexity with respect to the new representation.

We also derive lower bounds, showing that the dependence of our label-complexity guarantee on the accuracy c, or the margin parameter γ, is indeed necessary and is not an artifact of our analysis. We do not know if the dependence of our bounds on d is tight. It should be noted that some of the most popular learning algorithms (e.g. SVM, Perceptron, and AdaBoost) rely on a large-margin assumption to derive dimension-independent sample complexity guarantees. In contrast, here we use the margin for computational reasons. Our approximation guarantee depends logarithmically on the margin parameter, while the sample complexities of SVM, Perceptron, and AdaBoost depend polynomially on the margin. Hence, we require a much smaller margin than these algorithms do. In a related work, Balcan et al. (2007) proposed an active learning algorithm with dimension-independent guarantees under a margin assumption. These guarantees hold for a restricted class of data distributions.

In the second part of this work, we compare the greedy approach to the mellow approach of CAL in the realizable case, both theoretically and experimentally. Our theoretical results show the following: (1) In the simple learning setting of thresholds on the line, our margin-based approach is preferable to the mellow approach when the true margin of the target hypothesis is large. (2) There exists a distribution in Euclidean space such that the mellow approach cannot achieve a significant improvement in label complexity over passive learning for halfspaces, while the greedy approach achieves such an improvement using more unlabeled examples. (3) There exists a pool in Euclidean space such that the mellow approach requires exponentially more labels than the greedy approach.

We further compare the two approaches experimentally, both on separable data and on data with small error. The empirical evaluation indicates that our algorithm, which can be implemented in practice, achieves state-of-the-art results. It further suggests that aggressive approaches can be significantly better than mellow approaches in some practical settings.

We show main results for the greedy algorithm in Section 2. The ALuMA algorithm is described in Section 3. Results and experiments for the aggressive and the mellow approaches are presented in Section 4. Full proofs and additional experiments are provided in the full version of this paper (Gonen et al., 2012).

2. Results for Greedy Active Learning

Let us first introduce some notation. Given a pool $X = \{x_1, \ldots, x_m\}$, where each instance x_i is associated with an unknown label $L(i) \in \{\pm 1\}$, the goal of the learner is to find the values $L(1), \ldots, L(m)$ using as few calls to the oracle L as possible. We assume that L is determined by a function h taken from a predefined hypothesis class $\mathcal{H}$. That is, $\exists h \in \mathcal{H}$ such that for all i, $L(i) = h(x_i)$. We denote this by $L \Lleftarrow h$. An active learning algorithm $\mathcal{A}$ obtains (X, L, T) as input, where T is the label budget of $\mathcal{A}$. We denote by $N(\mathcal{A}, h)$ the number of calls to L that $\mathcal{A}$ makes before outputting $(L(x_1), \ldots, L(x_m))$, under the assumption that $L \Lleftarrow h$. The worst-case label complexity of $\mathcal{A}$ is defined to be $c_{\mathrm{wc}}(\mathcal{A}) \overset{\mathrm{def}}{=} \max_{h \in \mathcal{H}} N(\mathcal{A}, h)$. We denote the optimal worst-case label complexity for the given pool by OPT. Formally, we define $\mathrm{OPT} = \min_{\mathcal{A}} c_{\mathrm{wc}}(\mathcal{A})$, where the minimum is taken over all possible active learners for the given pool.

For a given active learner, we denote by $V_t \subseteq \mathcal{H}$ the version space of an active learner after t queries. Formally, suppose that the active learning queried instances $i_1, \ldots, i_t$ in the first t iterations needed. Then

$$V_t = \{h \in \mathcal{H} \mid \forall j \in [t], h(x_{i_j}) = L(i_j)\}.$$

For a given pool example $x \in X$, denote by $V_{t,x}^{j}$ the version spaces that would result if the algorithm now queried x and received label j. Formally,

$$V_{t,x}^{j} = V_t \cap \{h \in \mathcal{H} \mid h(x) = j\}.$$

Given X and $\mathcal{H}$, we define, for each $h \in \mathcal{H}$, the equivalence class of h over $\mathcal{H}$, $[h] = \{h' \in \mathcal{H} \mid \forall x \in X,\, h(x) = h'(x)\}$. We consider a probability distribution P over $\mathcal{H}$ such that $P([h])$ is defined for all $h \in \mathcal{H}$. For brevity, we denote $P(h) = P([h])$. Let $p_{\min} = \min_{h \in \mathcal{H}} P(h)$.

Consider an active learning algorithm $\mathcal{A}$. For any $\alpha \geq 1$, $\mathcal{A}$ is *α-approximately greedy* with respect to a probability distribution P over $\mathcal{H}$, if at any iteration of the algorithm t, the query $x \in X$ chosen by $\mathcal{A}$ satisfies

$$\mathbb{E}_{h \sim P}[V_{t,x}^{h(x)}] \geq \frac{1}{\alpha} \max_{x \in X} \mathbb{E}_{h \sim P}[1 - P(V_{t,x}^{h(x)})],$$

and the algorithm returns a labeling consistent with V_T upon termination. Golovin & Krause (2010) have shown that any α-approximately greedy algorithm finds the correct labeling after at most $O(\alpha \cdot \mathrm{OPT} \log(1/p_{\min}))$ queries.

We start by showing that by slightly changing the policy of an approximately-greedy algorithm, we can achieve a better approximation factor whenever the true target hypothesis has a larger probability than $p_{\min}$. This can be done by allowing the algorithm to stop before it reaches a pure version space, and requiring that in this case, it would output the labeling which is most likely based on the current version space and the fixed probability distribution P.

We say that $\mathcal{A}$ *outputs an approximate majority vote* if whenever V_T is pure enough, the algorithm outputs the majority vote on V_T. Formally, we define this as follows.

Definition 1. *An algorithm $\mathcal{A}$ outputs a β-approximate majority vote for $\beta \in (\frac{1}{2}, 1)$ if whenever there exists a labeling $Z : X \to \{\pm 1\}$ such that $P_{h \sim P}[Z \Lleftarrow h \mid h \in V_T] \geq \beta$, $\mathcal{A}$ outputs Z.*

In the following theorem we provide the target-dependent label complexity bound, which holds for any approximate greedy algorithm that outputs an approximate majority vote. We give here a sketch of the proof idea. The full proof is provided in Gonen et al. (2012).

Theorem 1. *Let $X = \{x_1, \ldots, x_m\}$. Let $\mathcal{H}$ be a hypothesis class, and let P be a distribution over $\mathcal{H}$. Suppose that $\mathcal{A}$ is α-approximately greedy with respect to P. Further suppose that it outputs a β-approximate majority vote. If $\mathcal{A}$ is executed with input (X, L, T) where $L \Lleftarrow h \in \mathcal{H}$, then $\mathcal{A}$ outputs $L(1), \ldots, L(m)$ after $T \geq \alpha(2\ln(1/P(h)) + \ln(\frac{\beta}{1-\beta})) \cdot \mathrm{OPT}$.*

Sketch. Fix a pool X. For any algorithm alg, denote by $V_t(\mathrm{alg}, h)$ the version space induced by the first n labels it queries if the true labeling of the pool is consistent with h. Denote the average version space reduction of alg after t queries by

$$f_{\mathrm{avg}}(\mathrm{alg}, t) = 1 - \mathbb{E}_{h \sim P}[P(V_t(\mathrm{alg}, h))].$$

Golovin & Krause (2010) prove that since $\mathcal{A}$ is α-approximately greedy, for any pool-based algorithm alg, and for every $k, t \in \mathbb{N}$,

$$f_{\mathrm{avg}}(\mathcal{A}, t) \geq f_{\mathrm{avg}}(\mathrm{alg}, k)(1 - \exp(-t/\alpha k)). \qquad (1)$$

Let opt be an algorithm that achieves OPT. We show that for any hypothesis $h \in \mathcal{H}$ and any active learner alg, $f_{\mathrm{avg}}(\mathrm{opt}, \mathrm{OPT}) - f_{\mathrm{avg}}(\mathrm{alg}, t) \geq P(h)(P(V_t(\mathrm{alg}, h)) - P(h))$. Combining this with Equation (1) we conclude that if $\mathcal{A}$ is α-approximately greedy then

$$\frac{P(h)}{P(V_t(\mathcal{A}, h))} \geq \frac{P(h)^2}{\exp(-\frac{t}{\alpha \mathrm{OPT}}) + P(h)^2}.$$

This means that if $P(h)$ is large enough and we run an approximate greedy algorithm, then after a sufficient number of iterations, most of the remaining version space induces the correct labeling of the sample. Specifically, if $n \geq \alpha(2\ln(1/P(h)) + \ln(\frac{\beta}{1-\beta})) \cdot \mathrm{OPT}$, then $P(h)/P(V_t(\mathcal{A},h)) \geq \beta$. Since $\mathcal{A}$ outputs a β-approximate majority labeling from $V_t(\mathcal{A},h)$, $\mathcal{A}$ returns the correct labeling. $\square$

When $P(h) \gg p_{\min}$, the bound in Theorem 1 is stronger than the guarantee obtained by Golovin & Krause (2010). Importantly, such an improved approximation factor *cannot be obtained* for a general approximate-greedy algorithm, even in a very simple setting. Thus, we can conclude that some algorithmic change is necessary. To show this, consider the setting of *thresholds on the line*. In this setting, the domain of examples is $[0,1]$, and the hypothesis class $\mathcal{H}_{\text{line}}$ includes all the hypotheses which are positive if the example is larger than some threshold in $[0,1]$.

Theorem 2. *Consider pool-based active learning on $\mathcal{H}_{\text{line}}$, and assume that P on $\mathcal{H}_{\text{line}}$ selects h_c by drawing the value c uniformly from $[0,1]$. For any $\alpha > 1$ there exists an α-approximately greedy algorithm $\mathcal{A}$ such that for any $m > 0$ there exists a pool $X \subseteq [0,1]$ of size m, and a threshold c such that $P(h_c) = 1/2$, while the label-complexity of $\mathcal{A}$ for $L \Leftarrow h_c$ is $\frac{m}{\lceil \log(m) \rceil} \cdot \mathrm{OPT}$.*

Interestingly, this theorem does not hold for $\alpha = 1$, that is for the exact greedy algorithm. This follows from Theorem 4, which we state and prove in Section 4.

So far we have considered a general hypothesis class. We now discuss the class of halfspaces in $\mathbb{R}^d$,$\mathcal{W} = \{x \mapsto \mathrm{sgn}(\langle w, x\rangle) : w \in \mathbb{B}_1^d\}$, where $\mathbb{B}_1^d$ is the unit ball in $\mathbb{R}^d$. For simplicity, we will slightly overload notation and sometimes use w to denote the halfspace it determines. We fix the distribution P to be the one that selects a vector w uniformly from $\mathbb{B}_1^d$. The following lemma shows that our active learning algorithm for halfspaces has the desired properties described above with high probability.

Lemma 1. *If ALuMA is executed with confidence δ, then with probability $1-\delta$ over its internal randomization, ALuMA is 4-approximately greedy and outputs a 2/3-approximate majority vote. ALuMA is polynomial in the pool size, the dimension, and $\log(1/\delta)$.*

Combining the above lemma with Theorem 1 we immediately obtain that ALuMA's label complexity is $O(\log(1/P(h)) \cdot \mathrm{OPT})$. We can upper-bound $\log(1/P(h))$ using the familiar notion of *margin*: For any hypothesis $h \in \mathcal{W}$ defined by some $w \in \mathbb{R}^d$, let $\gamma(h)$ be the maximal margin of the labeling of X by h, namely $\gamma(h) = \max_{v:\|v\|=1} \min_{i\in[m]} h(x_i)\langle v, x_i\rangle/\|x_i\|$. We show that for all $h \in \mathcal{W}$, $P(h) \geq \left(\frac{\gamma(h)}{2}\right)^d$. From this and Lemma 1, we obtain the following corollary, which provides a guarantee for ALuMA that depends on the margin of the target hypothesis.

Corollary 1. *Let $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{B}_1^d$, where $\mathbb{B}_1^d$ is the unit Euclidean ball of $\mathbb{R}^d$. Let $\delta \in (0,1)$ be a confidence parameter. Suppose that ALuMA is executed with input (X, L, T, δ), where $L \Leftarrow h \in \mathcal{W}$ and $T \geq 4(2d\ln(2/\gamma(h)) + \ln(2)) \cdot \mathrm{OPT}$. Then, with probability of at least $1-\delta$ over ALuMA's own randomization, it outputs $L(1), \ldots, L(m)$.*

Our result for ALuMA provides a target-dependent approximation factor guarantee, depending on the margin of the target hypothesis.[1] We can also consider the minimal possible margin, $\gamma = \min_{h\in\mathcal{W}} \gamma(h)$, and deduce from Theorem 1, or from the results of Golovin & Krause (2010), a uniform approximation factor of $O(d\log(1/\gamma))$. How small can γ be? The following result bounds this minimal margin from below under the reasonable assumption that the examples are represented by numbers of a finite accuracy.

Lemma 2. *Let $c > 0$ be such that $1/c$ is an integer and suppose that $X \subset \{-1, -1+c, \ldots, 1-c, 1\}^d$. Then, $\min_{h\in\mathcal{W}} \gamma(h) \geq (c/\sqrt{d})^{d+2}$.*

The proof is an adaptation of a classic result due to Muroga et al. (1961). We conclude that under this assumption for halfspaces, $p_{\min} = \Omega((c/d)^{d^2})$, and deduce an approximation factor of $d^2\log(d/c)$ for the worst-case label complexity of ALuMA. The exponential dependence of the minimal margin on d here is necessary; as shown in Håstad (1994), the minimal margin can indeed be exponentially small, even if the points are taken only from $\{\pm 1\}^d$.

We also derive a lower bound, showing that the dependence of our bounds on γ or on c is necessary. Whether the dependence on d is also necessary is an open question for future work.

Theorem 3. *For any $\gamma \in (0, 1/8)$, there exists a pool $X \subseteq \mathbb{B}_1^2 \cap \{-1, 1+c, \ldots, 1-c, 1\}^2$ for $c = \Theta(\gamma)$, and a target hypothesis $h^* \in \mathcal{W}$ for which $\gamma(h^*) = \Omega(\gamma)$, such that there exists an exact greedy algorithm that requires $\Omega(\ln(1/\gamma)) = \Omega(\ln(1/c))$ labels in order to output a correct majority vote, while the optimal algorithm requires only $O(\log(\log(1/\gamma)))$ queries.*

3. The ALuMA algorithm

We now describe our algorithm, listed below as Alg. 1, and explain why Lemma 1 holds. We name the algorithm *Active Learning under a Margin Assumption* or ALuMA. Its inputs are the unlabeled sample X, the

[1]One may suggest that the approximation factor we achieve for ALuMA in Lemma 1 is due to its internal randomization. We show in Gonen et al. (2012) that this is not the case.

labeling oracle L, the maximal allowed number of label queries T, and the desired confidence $\delta \in (0, 1)$. It returns the labels of all the examples in X.

Recall that we take P to be uniform over $\mathcal{W}$, the class of homogeneous half-spaces in $\mathbb{R}^d$. In each iteration we wish to choose, among the instances in the pool, the instance whose label would lead to the maximal expected reduction in the version space. Denote by I_t the set of indices corresponding to the elements in the pool whose label was not queried yet ($I_0 = [m]$). This is equivalent to finding, on round t, $k \overset{\text{def}}{=} \text{argmax}_{i \in I_t} P(V^1_{t,x_i}) \cdot P(V^{-1}_{t,x_i})$.

In order to be able to find k, we need to calculate the volumes of the sets $V^1_{t,x}$ and $V^{-1}_{t,x}$ for every element x in the pool. Both of these sets are convex sets obtained by intersecting the unit ball with halfspaces. The problem of calculating the volume of such convex sets in $\mathbb{R}^d$ is #P-hard if d is not fixed (Brightwell & Winkler, 1991). Moreover, deterministically approximating the volume is NP-hard in the general case (Matoušek, 2002). Luckily, it is possible to approximate this volume using randomization. Specifically, in Kannan et al. (1997) a randomized algorithm with the following guarantees is provided:

Lemma 3. *Let $K \subseteq \mathbb{R}^d$ be a convex body with an efficient separation oracle. There exists a randomized algorithm, such that given $\epsilon, \delta > 0$, with probability at least $1 - \delta$ the algorithm returns a number $\Gamma \geq 0$ such that $(1 - \epsilon)\Gamma < P(K) < (1 + \epsilon)\Gamma$. The running time of the algorithm is polynomial in $d, 1/\epsilon, \ln(1/\delta)$.*

We denote an execution of this algorithm on a convex body K by $\Gamma \leftarrow \text{VolEst}(K, \epsilon, \delta)$. The algorithm is polynomial in $d, 1/\epsilon, \ln(1/\delta)$. ALuMA uses this algorithm to estimate $P(V^1_{t,x})$ and $P(V^{-1}_{t,x})$ with sufficient accuracy. We denote these approximations by $\hat{v}_{x,1}$ and $\hat{v}_{x,-1}$ respectively. Using Lemma 3 we show that w.p. at least $1 - \delta/2$, ALuMA is 4-approximately greedy.

After T iterations, ALuMA needs to output the majority vote of a version space V that has a high enough purity level. To do this, we would like to uniformly draw several hypotheses from V and label X according to a majority vote over these hypotheses. This can be approximated using the hit-and-run algorithm (Lovász, 1999), which efficiently draws a random sample from a convex body $K \subseteq \mathbb{R}^d$, according to a distribution which is λ-close in total variation distance to the uniform distribution over K, in time $\tilde{O}(d^3/\lambda^2)$. In the final step of ALuMA, we produce a majority vote classification from a distribution which is $\frac{1}{12}$-close to uniform. This allows proving that ALuMA outputs a 2/3-approximate majority vote w.p. at least $1 - \delta/2$.

Algorithm 1 The **ALuMA** algorithm

```
1: Input: $X = \{x_1, \ldots, x_m\}$, $L : [m] \to \{-1, 1\}$, $T$, $\delta$
2: $I_1 \leftarrow [m]$, $V_1 \leftarrow \mathbb{B}^d_1$
3: for $t = 1$ to $T$ do
4:    $\forall i \in I_t, j \in \{\pm 1\}$, $\hat{v}_{x_i,j} \leftarrow \text{VolEst}(V^j_{t,x_i}, \frac{1}{3}, \frac{\delta}{4mT})$
5:    Select $i_t \in \text{argmax}_{i \in I_t}(\hat{v}_{x_i,1} \cdot \hat{v}_{x_i,-1})$
6:    $I_{t+1} \leftarrow I_t \setminus \{i_t\}$
7:    Request $y = L(i_t)$
8:    $V_{t+1} \leftarrow V_t \cap \{w : y\langle w, x_{i_t}\rangle > 0\}$
9: end for
10: $M \leftarrow \lceil 72 \ln(2/\delta) \rceil$.
11: Draw $w_1, \ldots, w_M$ $\frac{1}{12}$-uniformly from $V_{T+1}$.
12: For each $x_i$ return $y_i = \text{sgn}\left(\sum_{j=1}^{M} \text{sgn}(\langle w_j, x_i\rangle)\right)$.
```

3.1. Non-Separable Data and Kernels

If the data pool X is not separable, but a small upper bound on the total hinge-loss of the best separator can be assumed, then ALuMA can be applied after a preprocessing step. This preprocessing step maps the points in X to a set of points in a higher dimension, which are separable using the original labels of X. The dimensionality depends on the margin and on the bound on the total hinge-loss of the original representation. The preprocessing step can be enhanced also to support kernel representations, so that the original X can be represented by a kernel matrix as well. Applying ALuMA after this preprocessing steps results in an approximately optimal label complexity, however OPT here is measured with respect to the new representation. See the full details in Gonen et al. (2012). We demonstrate that in practice, this procedure provides good results on real data sets. Investigating the relationship between OPT in the new representation and OPT in the original representation is an important question for future work.

4. Other Approaches: A Theoretical and Empirical Comparison

We now compare the effectiveness of the approach of ALuMA to other active learning strategies. ALuMA can be characterized by two properties: (1) its "objective" is to reduce the volume of the version space; (2) at each iteration, it aggressively selects an example from the pool so as to (approximately) minimize its objective as much as possible (in a greedy sense). We discuss the implications of these properties by comparing to other strategies. Property (1) is contrasted with strategies that focus on increasing the number of examples whose label is known. Property (2) is contrasted with strategies which are "mellow", in that their criterion for querying examples is softer.

Much research has been devoted to the challenge of obtaining a substantial guaranteed improvement of label complexity over regular "passive" learning for halfspaces in $\mathbb{R}^d$. Examples (for the realizable case) include QBC (Seung et al., 1992; Freund et al., 1997), CAL (Cohn et al., 1994), and Active Perceptron (Dasgupta et al., 2005). These algorithms are not "pool-based" but rather use "selective-sampling": they sample one example at each iteration, and immediately decide whether to ask for its label. Out of these algorithms, CAL is the most mellow, since it queries any example whose label is yet undetermined by the version space. Its "objective" can be described as reducing the number of examples which are labeled incorrectly, since it has been shown to do so in many cases (Hanneke, 2007; 2011; Friedman, 2009). QBC and the Active Perceptron are less mellow. Their "objective" is similar to that of ALuMA since they decide on examples to query based on geometric considerations. In Section 4.1 we discuss the theoretical advantages and disadvantages of different strategies, by considering some interesting cases from a theoretical perspective. In Section 4.2 we report an empirical comparison of several algorithms and discuss our conclusions.

4.1. Theoretical Comparison

The label complexity of the algorithms mentioned above is usually analyzed in the PAC setting, thus we translate our guarantees into the PAC setting as well for the sake of comparison. We define the (ϵ, m, D)-label complexity of an active learning algorithm to be the number of *label queries* that are required in order to guarantee that given a sample of m unlabeled examples drawn from D, the error of the learned classifier will be at most ϵ (with probability of at least $1-\delta$ over the choice of sample). A pool-based active learner can be used to learn a classifier in the PAC model by sampling a pool of m unlabeled examples from D, applying the pool-based active learner to this pool, and running a passive learner on the labeled pool to obtain a classifier. For the class of halfspaces, if we sample an unlabeled pool of $m = \tilde{\Omega}(d/\epsilon)$ examples, the learned classifier will have an error of at most ϵ (with high probability).

To demonstrate the effect of property (1) presented above, consider again the case of thresholds on the line defined in Section 2. Compare two greedy pool-based active learners for $\mathcal{H}_{\text{line}}$: The first follows a binary search procedure, greedily selecting the example that increases the number of known labels the most. This requires $\lceil \log(m) \rceil$ queries to identify the correct labeling of the pool. The second algorithm queries the example that splits the version space as evenly as possible. Theorem 1 implies a label complexity of $O(\log(m)\log(1/\gamma(h)))$ for such an algorithm, since $\text{OPT}_{\max} = O(\log(m))$. However, a better result holds:

Theorem 4. *In the problem of thresholds on the line, for any pool with labeling L, the exact greedy algorithm, and any approximate greedy algorithm that outputs a majority vote, require at most $O(\log(1/\gamma(h)))$ labels.*

Comparing the $\lceil \log(m) \rceil$ guarantee of the first algorithm to the $\log(1/\gamma(h))$ guarantee of the second, we reach the (unsurprising) conclusion, that the first algorithm is preferable when the true labeling has a small margin, while the second is preferable when the true labeling has a large margin. This simple example accentuates the implications of selecting the volume of the version space as an objective. A similar implication can be derived in the PAC setting, when comparing CAL to ALuMA, with $m = \tilde{\Theta}(1/\epsilon)$. When $d = 1$, CAL achieves a label-complexity of $O(\log(1/\epsilon)) = O(\log(m))$, similarly to the binary search strategy. Thus when ϵ is large compared to $\gamma(h)$, CAL is better than being greedy on the volume, and the opposite holds when the condition is reversed. QBC will behave similarly to ALuMA in this setting.

To demonstrate the effect of property (2) described above—being aggressive versus being mellow—we consider an example adapted from Dasgupta (2006). In this example the distribution is supported by two circles in $\mathbb{R}^3$, one around the origin and one slightly above it. Dasgupta has demonstrated via this example that active learning can gain in label complexity from having significantly more unlabeled data. The following theorem shows that the aggressive strategy employed by ALuMA indeed achieves an exponential improvement when there are more unlabeled samples. In many applications, unlabeled examples are virtually free to sample, thus it can be worthwhile to allow the active learner to sample more examples than the passive sample complexity and use an aggressive strategy. In contrast, the mellow strategy of CAL does not significantly improve over passive learning in this case. We note that these results hold for any selective-sampling method that guarantees an error rate similar to passive ERM given the same sample size. This falls in line with the observation of Balcan et al. (2007), that in some cases a more aggressive approach is preferable.

Theorem 5. *For all small enough $\epsilon \in (0, 1)$ there is a distribution D_ϵ of points in $\mathbb{R}^3$, such that*

1. *For $m = O(1/\epsilon)$, the $(\epsilon, m, D_\epsilon)$-label complexity of* any active learner *is $\Omega(1/\epsilon)$.*

2. *For* $m = \Omega(\log^2(1/\epsilon)/\epsilon^2)$*, the* $(\epsilon, m, D_\epsilon)$*-label complexity of ALuMA is* $O(\log^2(1/\epsilon))$*.*

3. *For* any value *of* m*, the* $(\epsilon, m, D_\epsilon)$*-label complexity of CAL is* $\Omega(1/\epsilon)$*.*

This theorem demonstrates a case where more unlabeled examples can help ALuMA use less labels, whereas they do not help CAL. In fact, in some cases the label complexity of CAL can be significantly worse than that of the optimal algorithm, even when both CAL and the optimal algorithm have access to all the points in the support of the distribution. An example is provided in Gonen et al. (2012).

4.2. Empirical Comparison

We now report an empirical comparison between aggressive and mellow algorithms. Our goal is twofold: First, to evaluate ALuMA in practice, and second, to compare the performance of aggressive strategies and mellow strategies. The aggressive strategy is represented by ALuMA and by Tong & Koller (2002). The mellow strategy is represented by CAL. QBC represents a middle-ground between aggressive and mellow. We also compare to a passive ERM algorithm—one that uses random labeled examples.

Our implementation of ALuMA uses hit-and-run samples instead of full-blown volume estimation. See Gonen et al. (2012) for full details. QBC was also implemented using hit-and-run (Gilad-Bachrach et al., 2005). CAL and QBC were provided with a random ordering of the pool. The algorithm TK is the first heuristic proposed in Tong & Koller (2002), in which at each iteration the example closest to the max-margin solution of the labeled examples is selected. Since the active learners operate by reducing the training error, the graphs below compare the achieved training errors.

In all algorithms, we classify the training examples using the version space defined by the queried labels. The theory for CAL and ERM allows selecting an arbitrary predictor from the version space. In QBC, the predictor should be drawn uniformly at random from the version space. We found that all the algorithms perform significantly better if the majority vote classification proposed for ALuMA is used for them. Therefore, we used this methodology in for all algorithms.

First, we conducted a synthetic experiment on a sample from the uniform distribution on a sphere. Figure 1 depicts the training error as a function of the label budget when learning a random halfspace over the uniform distribution in $\mathbb{R}^{100}$. ALuMA and TK outperform CAL and QBC here. In Gonen et al. (2012),

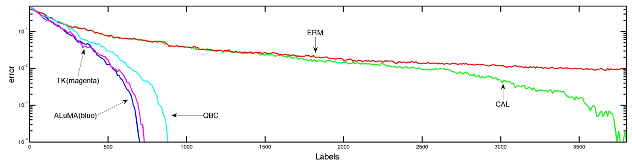

Figure 1. Uniform distribution ($d = 100$).

we show the behavior in $\mathbb{R}^{10}$ as well. The difference between the performance of the different algorithms is less marked in the lower dimension, suggesting that the difference might grow with the dimension. These results indicate that ALuMA might have a better guarantee than the general relative analysis in the case of the uniform distribution. Achieving such an analysis is an open question which is left for future work.

We next tested MNIST,[2] which depicts gray-scale images of digits in dimension 784. Figure 2(left) shows the training error as a function of the label budget for learning to distinguish between the digits 3 and 5. Strikingly, CAL provides no improvement over passive ERM in the first 1000 examples, ALuMA and TK reach zero training error. We tested also PCMAC,[3] after a random projection from the original dimension of 7511 to dimension 300. The results are shown in Figure 2(right). Again, CAL does not improve over passive ERM unlike the aggressive approaches.

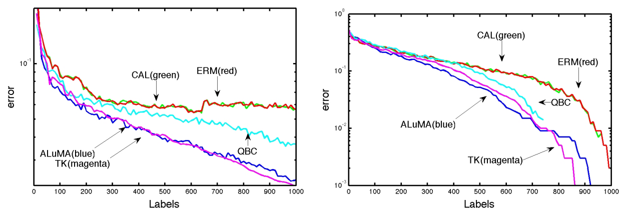

Figure 2. MNIST (3 vs. 5) (left) and PCMAC (right)

In the experiments reported so far, TK and ALuMA perform about the same, showing that the TK heuristic is very successful. However, there are cases where TK performs much worse than ALuMA. We report a relevant synthetic experiment in Gonen et al. (2012). We also present experiments on non-separable data sets, showing that when the error is low and an upper bound on the hinge-loss can be assumed, ALuMA can improve performance over mellow approaches.

Acknowledgement : This work was supported by the Germani-Israeli Foundation grant number 0378590.

[2]http://yann.lecun.com/exdb/mnist/

[3]http://vikas.sindhwani.org/datasets/lskm/matlab/

References

Arkin, E.M., Meijer, H., Mitchell, J.S.B., Rappaport, D., and Skiena, S.S. Decision trees for geometric models. In *Proceedings of the ninth annual symposium on Computational geometry*, pp. 369–378. ACM, 1993.

Balcan, M.F., Beygelzimer, A., and Langford, J. Agnostic active learning. In *Proceedings of the 23rd international conference on Machine learning*, pp. 65–72. ACM, 2006.

Balcan, M.F., Broder, A., and Zhang, T. Margin based active learning. *Learning Theory*, pp. 35–50, 2007.

Brightwell, G. and Winkler, P. Counting linear extensions is #p-complete. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, STOC '91, pp. 175–181, 1991.

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

Dasgupta, S. Analysis of a greedy active learning strategy. *Advances in neural information processing systems*, 17: 337–344, 2005.

Dasgupta, S. Coarse sample complexity bounds for active learning. *Advances in neural information processing systems*, 18:235, 2006.

Dasgupta, S., Kalai, A., and Monteleoni, C. Analysis of perceptron-based active learning. *Learning Theory*, pp. 889–905, 2005.

Dasgupta, S., Hsu, D., and Monteleoni, C. A general agnostic active learning algorithm. *Advances in neural information processing systems*, 20:353–360, 2007.

El-Yaniv, R. and Wiener, Y. Active learning via perfect selective classification. *The Journal of Machine Learning Research*, 13:255–279, 2012.

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

Friedman, E. Active learning for smooth problems. In *Proceedings of the 22nd Conference on Learning Theory*, volume 1, pp. 3–2, 2009.

Gilad-Bachrach, R., Navot, A., and Tishby, N. Query by committee made real. *Advances in Neural Information Processing Systems (NIPS)*, 19, 2005.

Golovin, D. and Krause, A. Adaptive submodularity: A new approach to active learning and stochastic optimization. In *Proceedings of International Conference on Learning Theory (COLT)*, 2010.

Gonen, A., Sabato, S., and Shalev-Shwartz, S. Efficient pool-based active learning of halfspaces. *CoRR*, abs/1208.3561, 2012.

Hanneke, S. A bound on the label complexity of agnostic active learning. In *ICML*, 2007.

Hanneke, S. Rates of convergence in active learning. *The Annals of Statistics*, 39(1):333–361, 2011.

Håstad, J. On the size of weights for threshold gates. *SIAM Journal on Discrete Mathematics*, 7:484, 1994.

Kannan, R., Lovász, L., and Simonovits, M. Random walks and an $o*(n^5)$ volume algorithm for convex bodies. *Random structures and algorithms*, 11(1):1–50, 1997.

Lovász, L. Hit-and-run mixes fast. *Mathematical Programming*, 86(3):443–461, 1999.

Matoušek, J. *Lectures on discrete geometry*, volume 212. Springer Verlag, 2002.

McCallum, A. and Nigam, K. Employing em in pool-based active learning for text classification. In *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pp. 350–358, 1998.

Muroga, S., Toda, I., and Takasu, S. Theory of majority decision elements. *Journal of the Franklin Institute*, 271 (5):376–418, 1961.

Seung, H.S., Opper, M., and Sompolinsky, H. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 287–294. ACM, 1992.

Tong, S. and Koller, D. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.