Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Efficient Active Learning of Halfspaces via Query Synthesis

Ibrahim Alabdulmohsin and **Xin Gao** and **Xiangliang Zhang**
Computer, Electrical and Mathematical Sciences & Engineering Division
King Abdullah University of Science & Technology (KAUST)
Thuwal, Saudi Arabia 23955

Abstract

Active learning is a subfield of machine learning that has been successfully used in many applications including text classification and bioinformatics. One of the fundamental branches of active learning is *query synthesis*, where the learning agent constructs artificial queries from scratch in order to reveal sensitive information about the true decision boundary. Nevertheless, the existing literature on membership query synthesis has focused on finite concept classes with a limited extension to real-world applications. In this paper, we present an efficient spectral algorithm for membership query synthesis for halfspaces, whose sample complexity is experimentally shown to be near-optimal. At each iteration, the algorithm consists of two steps. First, a convex optimization problem is solved that provides an approximate characterization of the version space. Second, a principal component is extracted, which yields a synthetic query that shrinks the version space exponentially fast. Unlike traditional methods in active learning, the proposed method can be readily extended into the batch setting by solving for the top k eigenvectors in the second step. Experimentally, it exhibits a significant improvement over traditional approaches such as uncertainty sampling and representative sampling. For example, to learn a halfspace in the Euclidean plane with 25 dimensions and an estimation error of 1E-4, the proposed algorithm uses less than 3% of the number of queries required by uncertainty sampling.

Introduction

Active learning is a subfield of machine learning that has received a growing interest over the past 20 years. It has been successfully used in many applications including character recognition, text classification, computational chemistry, spam filtering, and bioinformatics (Liu 2004; Sculley 2007; Settles 2010). In addition, many software companies are increasingly reliant on active learning techniques, such as Google, IBM, and CiteSeer (Settles 2010; 2011).

The central goal of active learning (also called "query learning") is to be able to learn well using only *a few* queries or training examples. There are various scenarios in which active learning arises quite naturally. For instance, in speech recognition and text classification, there is often an abundance of *unlabeled* examples that can be used to improve classification accuracy. However, the labeling cost is not negligible, and the goal is to nominate a small subset of such unlabeled examples for manual human annotation. Similarly, active learning can be used in dataset compression, in which the entire training set is reduced in size in order to improve the training time.

More precisely, suppose throughout this paper that our instance space is $\mathbb{R}^d$, and suppose we would like to estimate a halfspace $c^\star = \{x \in \mathbb{R}^d \,|\, \langle w^\star,\, x\rangle \geq 0\}$ for some unknown normal vector $w^\star \in \mathbb{R}^d$. The goal of active learning is to be able to estimate $c^\star$, or equivalently $w^\star$, using as few queries as possible. Here, a query comprises of an instance $x_i \in \mathbb{R}^d$ and a membership response:

$$y_i = c^\star(x_i) = \mathbb{I}\{x_i \in c^\star\} = \textbf{sign}(\langle x_i,\, w^\star\rangle) \in \{+1,\, -1\}$$

We assume we operate in the *realizable* setting. That is, we have for all queries $\{(x_1,\, y_1),\, (x_2,\, y_2), \ldots\}$:

$$\exists w \in \mathbb{R}^d \,:\, \forall i, y_i = \textbf{sign}(\langle x_i,\, w\rangle)$$

In addition, we assume that the true halfspace $c^\star$ (a.k.a. the target concept) is deterministic and noise-free. These assumptions are widely used in the literature (see for instance (Settles 2010; Balcan and Long 2013; Tong and Koller 2002; Brinker 2003; Balcan, Broder, and Zhang 2007; Dasgupta 2005b; Freund et al. 1997; Dasgupta, Kalai, and Monteleoni 2009)). They are satisfied for many important classification problems such as text classification (Xu et al. 2003). In addition, the assumed setting is equivalent to the task of *reverse-enigneering* linear classifiers.

The traditional Probably Approximately Correct (PAC) model (Valiant 1984; Haussler and Warmuth 1993) can be used to provide sample complexity bounds for learning halfspaces. Specifically, suppose that $\mathcal{D}$ is a probability distribution of instances $x \sim \mathcal{D}$ and let $\hat{c}$ be the concept estimated according to the training set $\{(x_1,\, y_1),\, (x_2,\, y_2), \ldots\}$, whose instances x_i are drawn i.i.d. according to $\mathcal{D}$ while $y_i = c^\star(x_i) \in \{+1,\, -1\}$. Define estimation error to be $\epsilon = \mathbb{P}_{x\sim\mathcal{D}}[\hat{c}(x) \neq c^\star(x)]$. Then, a classical result using the PAC model states that $\tilde{O}(\frac{d}{\epsilon})$ training examples are needed in order to achieve an estimation error of ϵ (Cohn, Atlas, and Ladner 1994; Dasgupta 2005a; Balcan, Broder, and Zhang 2007; Balcan and Long 2013). The hope of active learning is to

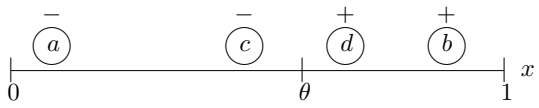

Figure 1: Using the bisection search, query learning of thresholds in $\mathbb{R}$ can yield exponential improvement in sample complexity over random sampling.

be able to reduce this sample complexity exponentially fast into $\tilde{O}(d \log \frac{1}{\epsilon})$. A sample complexity of $\Omega(d \log \frac{1}{\epsilon})$ can be shown to be optimal using a sphere counting argument (Dasgupta, Kalai, and Monteleoni 2009).

It is easy to observe how active learning helps achieve such dramatic reduction in sample complexity when $d = 1$ (Dasgupta 2005b). Consider the example depicted in Figure 1, where our target concept is $c^\star = \{x \in \mathbb{R} \,|\, x \geq \theta\}$ for some unknown $\theta \in [0, 1]$, and our instances x_i are uniformly distributed in the unit interval $[0, 1]$. Using the bisection method, one can estimate θ very quickly. If we start with the two examples $(x_1 = a, y_1 = -1)$ and $(x_2 = b, y_2 = +1)$, then the next *synthetic* queries would be $(x_3 = c, y_3 = -1)$ and $(x_4 = d,\ y_4 = +1)$ in tandem. Clearly, each query cuts the region of uncertainty by one half regardless of its label. Hence, we can achieve an error rate of ϵ using only $\tilde{O}(\log \frac{1}{\epsilon})$ queries (Dasgupta 2005b).

Extending the bisection method into $\mathbb{R}^d$ for all $d \geq 1$ is possible, at least in principle, using what is commonly referred to as the "halving algorithm" (Tong and Koller 2002; Brinker 2003). Unfortunately, the halving algorithm is an *idealized* method that is often prohibitively complex to implement in practice (Cohn, Atlas, and Ladner 1994; Tong and Koller 2002; Settles 2010). It operates by first computing the set of all hypotheses that are consistent with all past queries. This set of possible hypotheses is referred to as the *version space*, a term that was first introduced by Tom Mitchell (Mitchell 1982). After that, a query is synthesized that cuts the "volume" of the version space by one half. Then, the new version space is computed and the entire process is repeated multiple times. The halving algorithm can be justified rigorously (Angluin 2001; Tong and Koller 2002; Dasgupta 2005a; 2005b).

Because the ideal halving algorithm is often difficult to implement in practice, *pool-based* approximations are used instead such as uncertainty sampling and the query-by-committee (QBC) algorithm (Freund et al. 1997; Tong and Koller 2002; Brinker 2003). Unfortunately, such approximation methods are only guaranteed to work well if the number of unlabeled examples (i.e. pool size) grows *exponentially* fast with each iteration. This is due to the fact that the required pool size is always proportional to $\tilde{O}(\frac{1}{\epsilon})$ (Freund et al. 1997; Balcan, Broder, and Zhang 2007; Dasgupta, Kalai, and Monteleoni 2009). Otherwise, such heuristics become crude approximations and they can perform quite poorly (Tong and Koller 2002; Schein and Ungar 2007). Because the required pool size grows exponentially fast before we can guarantee an exponential reduction in sample complexity, *query synthesis* offers an attractive alternative approach. In query synthesis, new queries are constructed *de novo*, and hence the pool size limitation is completely eliminated.

In this paper, we provide an efficient membership query synthesis algorithm for halfspaces. The new algorithm can also be interpreted as an approximation to the ideal halving algorithm. Nonetheless, we retain its key advantage; namely that its estimation error enjoys an exponential improvement over random sampling. At each iteration, the new algorithm consists of two steps. First, a convex optimization problem is solved that provides an approximate characterization of the version space. In the second step, a principal component is extracted, which yields the optimal synthetic query that shrinks the version space exponentially fast. Both steps can be implemented quite efficiently.

In addition to the classical active learning setting in which a *single* query is synthesized at each iteration, the proposed algorithm can also be readily extended into the *batch* setting, where $k \geq 1$ queries are synthesized per iteration. Batch-mode active learning has been recently described as one of the *top challenges* in active learning (Settles 2011). It is important in distributed parallel labeling environments, such as in biology (Settles 2010). Ideally, batch-mode active learning should produce queries that are both informative and diverse (Brinker 2003). In our case, the new proposed algorithm operates in the batch setting by solving for the top k eigenvectors in the second step of each iteration, thus achieving both objectives (details are provided in the sequel).

The rest of the paper is structured as follows. First, we present a statement of the problem and discuss related work. After that, we derive the new query synthesis algorithm, and show how it can serve as an approximation to the ideal halving algorithm. Next, we demonstrate that the proposed method indeed achieves an exponential reduction in sample complexity compared to random sampling, and that its performance is significantly superior to popular active learning techniques.

Problem Statement

Halfspaces are generally considered to be one of the most important concept classes in practice today (Fan et al. 2008; Yuan, Ho, and Lin 2012; Balcan and Long 2013). The class of linear separators is broad and includes support vector machines (SVM), the perceptron, as well as logistic regression, and they can achieve comparable accuracy to non-linear classifiers in many applications (Yuan, Ho, and Lin 2012). In fact, due to their wide popularity, many solvers currently exist for linear classification including SVMperf, OCAS, Pegasos, SGD, and LIBLINEAR (Yuan, Ho, and Lin 2012). In this paper, we present a new efficient algorithm for active learning of halfspaces via query synthesis.

There is an emerging class of new applications for which active learning via query synthesis is a useful tool. These include automated science, such as the "robot scientist" experiment described in (King and others 2009), and adversarial reverse engineering (Nelson et al. 2012). For applications in which we desire to learn a halfspace with arbitrary accuracy, pool-based query learning strategies such as

uncertainty sampling and QBC are inefficient because the required pool size grows exponentially fast with each iteration. Query synthesis, by contrast, builds new queries *de novo* and, hence, offers an attractive alternative approach.

In the literature, query synthesis strategies have focused on either finite concept classes or on toy problems such as geometrical shapes in two dimensions (Angluin 2001; Settles 2010). In this paper, we aim at deriving a new query synthesis algorithm for learning halfspaces in $\mathbb{R}^d$ for all $d \geq 1$. We require that complexity of the query synthesis algorithm to grow only *polynomially* with each iteration, while still offering an *exponential* reduction in estimation error.

Related Work

There are two fundamental branches of active learning proposed in the literature: (1) query synthesis, and (2) pool-based sampling (Settles 2010). In the query synthesis approach, the learning agent constructs new queries *de novo* in order to reveal sensitive information about the true decision boundary. For instance, an autonomous robot scientist might infer hypotheses based on observations, and design new queries in order to refine those hypotheses (King and others 2009). Query synthesis has traditionally focused on regression tasks, under the name of *optimal experiemental design*, because the optimal query can often be determined analytically in such cases (Freund et al. 1997; Settles 2010). However, it has seldom been extended into the classification setting except for finite concept classes or for artificial toy problems (Angluin 2001; Settles 2010).

Pool-based sampling (Lewis and Gale 1994), by contrast, has been successfully applied for a wide range of problems including classification and regression (Schein and Ungar 2007; Settles 2010). In this setting, it is assumed that a large pool of unlabeled examples is available, hence the name, and the goal is to select a small subset of those examples to serve as new queries. Perhaps, the two most dominant approaches in pool-based sampling are *uncertainty sampling* and the *query-by-committee* (QBC) algorithm (Schein and Ungar 2007; Settles 2011). Both can be interpreted as approximations to the ideal halving algorithm that was discussed earlier (Tong and Koller 2002; Freund et al. 1997). For the concept class of homogenous (i.e. through the origin) halfspaces, Balcan, Broder, and Zhang showed that uncertainty sampling could achieve an exponential reduction in sample complexity (Balcan, Broder, and Zhang 2007), while Dasgupta, Kalai, and Monteleoni proved similar performance guarantees for a perceptron-like uncertainty sampling algorithm (Dasgupta, Kalai, and Monteleoni 2009). Similarly, Freund et al. showed that the QBC algorithm could achieve similar performance for the same concept class of homogenous halfspaces (Freund et al. 1997). However, all the pool-based sampling methods above require a pool size that grows exponentially fast.

Moreover, numerous other algorithms have been proposed for pool-based sampling. These include algorithms that are based on the expected error reduction (Roy and McCallum 2001), bagging and boosting (Abe and Mamitsuka 1998), and expectation-maximization (EM) (McCallum and Nigam 1998). Nevertheless, uncertainty sampling and the QBC algorithm remain dominant and they perform quite competitively (Schein and Ungar 2007; Settles 2011).

In batch-mode active learning, on the other hand, very few algorithms have been previously proposed (Settles 2011). In principle, the goal is to introduce diversity among queries as a whole, while also ensuring that each query is informative by itself. In order to achieve both objectives, two general approaches have been proposed. The first approach is called *representative sampling*, which works by first clustering informative instances x_i into k groups, and picking the k representative samples for those groups as queries (Xu et al. 2003). The second approach is optimization-based, which treats diversity and uncertainty as two separate terms in a single objective function. For the concept class of homogenous halfspaces, diversity is measured by the pairwise angles between queries (Brinker 2003). That is, diversity is defined by orthogonality.

Notation

The notation used in this paper is fairly standard. Vectors will be denoted using small letters z while matrices will be denoted using capital letters Z. We will always denote instances by $x \in \mathbb{R}^d$ and membership labels by $y \in \{+1, -1\}$. Here, $y = c^\star(x)$ indicates whether or not x belongs to the target concept $c^\star$. We will use x^T to denote the transpose of x, and use $||x||_A$ to denote the induced norm of x by the positive semidefinite matrix A. That is, $||x||_A = \sqrt{x^T A\, x}$. We will write $A \succeq 0$ to mean that the matrix A is positive semidefinite. The cone of $d \times d$ symmetric positive semidefinite matrices will be denoted $\mathbb{S}^d_+$. Finally, we write $a \circ b$ to denote the Hadamard (element-wise) product between the two vectors a and b.

Query Synthesis for Halfspaces

Version Space Approximation

Before we derive a query synthesis algorithm for halfspaces, let us first recapitulate how the *ideal* halving algorithm works. The ideal halving algorithm works in the following manner:

- STEP I: Compute the version space $\mathcal{V}$, which is the set of all hypotheses that are consistent with all past queries $\{(x_i, y_i)\}_i$. Because we only need to recover the normal vector w up to a constant factor in halfspace learning, the version space can be characterized in terms of w by the intersection of the surface of the unit sphere with a polyhedral set:

$$\mathcal{V} = \{w \in \mathbb{R}^d \mid \; ||w||_2 = 1 \wedge \forall i,\; y_i \langle x_i,\, w \rangle \geq 0\} \quad (1)$$

- STEP II: Measure the "volume" of $\mathcal{V}$. One natural measure of volume is the surface area of the unit sphere that lies inside the polyhedral region as defined in Eq. (1). Stop if the volume of $\mathcal{V}$ is below some predefined tolerance.
- STEP III: Synthesize a new query that cuts the volume of $\mathcal{V}$ by one half. That is, find an instance x_{t+1} such that the volume of the *new* version space after observing query $(x_{t+1},\, y_{t+1})$ is reduced by one half regardless of the value of $y_{t+1} \in \{+1, -1\}$. Then, increment t and go back to STEP I.

The key difficulty in query synthesis that makes the above halving algorithm prohibitively complex lies in the nature of the version space. Because we can always assume that $||w||_2 = 1$, an ideal definition of the version space is given by Eq. (1) whose natural measure of volume is the surface area. This definition, however, is difficult to deal with analytically.

In order to circumvent such limitation, we propose to approximate the version space $\mathcal{V}$ by the largest ellipsoid $\varepsilon^\star = (\mu^\star, \Sigma^\star)$, with mean $\mu^\star$ and covariance $\Sigma^\star$, which is consistent with all past queries and whose mean $\mu^\star$ lies at the surface of the unit sphere, i.e. $||\mu^\star||_2 = 1$. This approximation can be interpreted as a *smoothening* of the original ideal definition of $\mathcal{V}$ so that query synthesis can be carried out quite efficiently.

In order to determine $\varepsilon^\star$, we need to determine its mean $\mu^\star$ and covariance matrix $\Sigma^\star$. By definition of $\epsilon^\star$, we have:

$$w \in \varepsilon^\star \quad \Leftrightarrow \quad \left|\left|w - \mu^\star\right|\right|_{\Sigma^{\star -1}} \leq 1 \tag{2}$$

To guarantee that the entire ellipsoid $\varepsilon^\star$ is consistent with a query (x_i, y_i), it can be shown (e.g. using Lagrange duality) that Eq. (2) implies that the following condition is both necessary and sufficient:

$$y_i \langle x_i, \mu^\star \rangle \geq ||x_i||_{\Sigma^\star} \tag{3}$$

The volume of an ellipsoid $\varepsilon = (\mu, \Sigma)$ with mean μ and covariance matrix Σ is proportional to $\det \Sigma$. Because the logarithmic function is monotone and increasing, maximizing $\det \Sigma$ is equivalent to maximizing the concave function $\log \det \Sigma$. Consequently, in order to solve for $\varepsilon^\star$, we need to solve the following optimization problem:

$$\begin{aligned} \underset{\mu,\ \Sigma \in \mathbb{S}^d_+}{\text{maximize}} \quad & \log \det \Sigma \\ \text{subject to} \quad & y_i \cdot (\mu^T x_i) \geq \sqrt{x_i^T \Sigma x_i}, \quad \text{for all } i = 1, \dots \\ & ||\mu||_2 = 1 \end{aligned} \tag{4}$$

The optimization problem, however, is not convex. To cast it as a convex optimization problem, so that it can be solved quite efficiently, we introduce the new variable $S = \Sigma^{\frac{1}{2}}$. Furthermore, we replace the equality constraint with an inequality constraint as follows:

$$\begin{aligned} \underset{\mu,\ S \in \mathbb{S}^d_+}{\text{maximize}} \quad & \log \det S \\ \text{subject to} \quad & y_i \cdot (\mu^T x_i) \geq ||S x_i||_2, \quad \text{for all } i = 1, 2, \dots \\ & ||\mu||_2 \leq 1 \end{aligned} \tag{5}$$

It is clear that the constraint $||\mu||_2 \leq 1$ holds with equality at optimality. Otherwise, we can multiply both μ and S by $||\mu||_2^{-1}$, which increases the objective function without violating the constraints. Therefore, the optimal solution to the optimization problem (5) is precisely the desired $\varepsilon^\star$.

The optimization problem in (5) is convex. Upon making the substitution $\Sigma = S^2$, we recover the optimal solution to the original optimization problem (4). Thus, $\varepsilon^\star$ can be determined efficiently, which provides us with a smoothed approximation to the version space.

In applications where $d \gg 1$, estimating the entire $d \times d$ covariance matrix in (5) is computationally expensive. One method to remedy such a problem is to assume that S is diagonal, which reduces the number of variables from $\Theta(d^2)$ down to $\Theta(d)$. This has the additional advantage of replacing the log-determinant function with a geometric-mean that can be handled more readily by *second-order cone programming* (SOCP) solvers (Lobo et al. 1998). The new optimization problem becomes:

$$\begin{aligned} \underset{\mu,\ s \geq 0}{\text{maximize}} \quad & \left(\prod_{j=1}^{d} s_j\right)^{\frac{1}{d}} \\ \text{subject to} \quad & y_i \cdot (\mu^T x_i) \geq ||s \circ x_i||_2, \quad \text{for all } i = 1, 2, \dots \\ & ||\mu||_2 \leq 1 \end{aligned} \tag{6}$$

Efficient Implementation

The optimization problem in (5) falls under the general class of *maximum-determinant* (max-det) optimization problems. This is a class of optimization problems that have been studied extensively in the literature. They can be solved quite efficiently, both in the worst-case complexity theory and in practice (Vandenberghe, Boyd, and Wu 1998).

On the other hand, the optimization problem in (6) can be cast as a SOCP problem by introducing new variables (Lobo et al. 1998). SOCP solvers are generally very efficient. Using CVX (CVX Research 2012), for instance, the optimization problem in (6) takes 2s, 4s, and 14s to solve when $d = 100$ and the number of queries is 1000, 2000, and 5000 respectively. Similarly, it takes 4s, 9s, and 35s to solve when $d = 200$ and with 1000, 2000, and 5000 queries respectively. The time complexity is almost linear with respect to the problem dimension d and to the number of queries.

Query Synthesis

Having obtained a smoothed approximation to the version space $\mathcal{V}$, the next step is to use our knowledge of $\varepsilon^\star$ to synthesize new queries. If our goal is to cut the volume of $\varepsilon^\star$ by one half, then any query that resides in the orthogonal complement of $\mu^\star$ will achieve such objective as proved next.

Lemma 1 (Uncertainty Sampling). *Suppose $\varepsilon^\star$ at iteration t has mean $\mu^\star$ and covariance matrix $\Sigma^\star$. Let x_{t+1} be any instance such that $\langle x_{t+1}, \mu^\star \rangle = 0$. Then, regardless of the label $y_{t+1} \in \{+1, -1\}$, exactly half of the points in $\varepsilon^\star$ are inconsistent with (x_{t+1}, y_{t+1}).*

Proof. In order to show that exactly half of the points in $\varepsilon^\star$ are inconsistent with (x_{t+1}, y_{t+1}), we need to show that a one-to-one mapping exists between the set of consistent points $\varepsilon^+ = \{w \in \varepsilon^\star \,|\, y_{t+1}\langle x_{t+1}, w\rangle > 0\}$ and the set of inconsistent points $\varepsilon^- = \{w \in \varepsilon^\star \,|\, y_{t+1}\langle x_{t+1}, w\rangle < 0\}$. We ignore the case $\langle x_{t+1}, w\rangle = 0$ because the volume of such a set is zero.

First, for any $w \in \mathbb{R}^d$, write $\bar{w} = \mu^\star - (w - \mu^\star)$. Note that $\bar{\bar{w}} = w$. Our first claim is that the condition $w \in \varepsilon^\star$ is equivalent to the condition $\bar{w} \in \varepsilon^\star$. This can be easily seen by using Eq. (2). If $w \in \varepsilon^\star$, then we have by definition:

$$(w - \mu^\star)\, \Sigma^{\star -1} (w - \mu^\star) \leq 1,$$

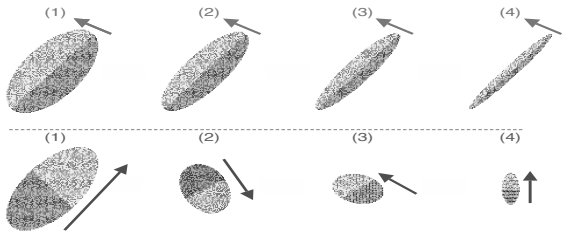

Figure 2: By blindly eliminating "half" of the hypotheses in $\varepsilon^\star$, we may push $\varepsilon^\star$ into a lower-dimensional subspace as depicted at the top, where arrows indicate the chosen direction of the cut. Here, the fact that volume$(\varepsilon^\star) \to 0$ does not imply that $\varepsilon^\star \to 0$. Instead, it is imperative to shrink both volume and axis length as depicted at the bottom.

which is also satisfied for $\bar{w}$. The converse also holds. Hence, the statements $w \in \varepsilon^\star$ and $\bar{w} \in \varepsilon^\star$ are equivalent.

Next, we show that for any query $(x_{t+1},\, y_{t+1})$ that satisfies the orthogonality condition $\langle x_{t+1},\, \mu^\star \rangle = 0$ and the condition $\langle x_{t+1},\, w \rangle \neq 0$, *exactly* one of w or $\bar{w}$ will be consistent with $(x_{t+1},\, y_{t+1})$ for any $w \in \varepsilon^\star$. This can be seen by writing:

$$\begin{aligned} y_{t+1} \langle x_{t+1},\, w \rangle &= y_{t+1} \langle x_{t+1},\, \mu^\star + (w - \mu^\star) \rangle \\ &= y_{t+1} \langle x_{t+1},\, w - \mu^\star \rangle \end{aligned}$$

In the second line, we used the orthogonality condition $\langle x_{t+1},\, \mu^\star \rangle = 0$. However, for $\bar{w} \in \varepsilon^\star$, we also have by definition of $\bar{w}$:

$$y_{t+1} \langle x_{t+1},\, \bar{w} \rangle = -y_{t+1} \langle x_{t+1},\, w - \mu^\star \rangle$$

Therefore, if $w \in \varepsilon^+$, then $\bar{w} \in \varepsilon^-$ and vice versa. Since $w,\, \bar{w} \in \varepsilon^\star$, we deduce the statement of the lemma. □

Lemma 1 reveals that *uncertainty sampling*, i.e. picking a random query that *exactly* lies in the orthogonal complement of $\mu^\star$, will always eliminate half of the points in $\varepsilon^\star$. Of course, this does *not* imply that the volume of $\varepsilon^\star$ at the next iteration will reduce by one half since an entirely different ellipsoid in an entirely different region might be picked next. However, by synthesizing a query that cuts the largest ellipsoid by one half, we are effectively approximating the ideal halving algorithm in a greedy fashion.

Unfortunately, uncertainty sampling alone is not very effective. This can be observed by noting that even if the volume of $\varepsilon^\star$ converges to zero, the ellipsoid itself may not converge to zero. For example, $\varepsilon^\star$ might be pushed towards a lower dimensional subspace. What we desire, instead, is to guarantee that all *axes* of the ellipsoid are shrank exponentially fast. As depicted in Figure 2, one method to accomplish this is to cut $\varepsilon^\star$ along its largest axes, where the weight of each axis is determined by its length.

Formally speaking, we would like to maximize the projection $\langle x_{t+1},\, v_j \rangle$, where v_j are the eigenvectors of $\Sigma^\star$. However, we do not care about the sign of $\langle x_{t+1},\, v_j \rangle$, i.e. we do not care if the light and dark regions in Figure 2 are swapped. In addition, each v_j is weighted by its length λ_j, where λ_j is the corresponding eigenvalue. These requirements lead to the following optimization problem:

$$\begin{aligned} \underset{x:\, ||x||_2=1}{\text{maximize}} \quad & \sum_{j=1}^{d} \lambda_j \langle x,\, v_j \rangle^2 \\ \text{subject to} \quad & \langle \mu^\star,\, x \rangle = 0 \end{aligned} \tag{7}$$

Because the norm of x_{t+1} does not matter in our queries, we fix $||\, x\, ||_2 = 1$ in (7). If we let N be the orthonormal basis of the orthogonal complement of $\mu^\star$ and write $x = N\alpha$, then the optimization problem (7) can be rewritten as:

$$\underset{\alpha\,:\,||\alpha||_2=1}{\text{maximize}} \quad \alpha^T (SN)^T (SN)\, \alpha \tag{8}$$

Recall here that $S = \Sigma^{\star\, \frac{1}{2}}$, which is the optimal solution to (5) or (6), depending on which formulation is used. Because N is orthonormal, the condition $||x||_2 = 1$ is equivalent to the condition $||\alpha||_2 = 1$.

It is an elementary result in linear algebra that the solution to the optimization problem (8), denoted as $\alpha^\star$, is the top eigenvector of the positive semidefinite matrix $N^T S^T S N = N^T \Sigma^\star\, N$, which can be computed efficiently. Therefore, we choose as our next query the unique instance:

$$x_{t+1} = N\, \alpha^\star \tag{9}$$

Batch-mode Query Synthesis

Extending the proposed query synthesis algorithm into the batch setting is straightforward. As suggested in the literature, we would like our k queries to be both informative and diverse. In our case, this amounts to the requirements that all k queries belong to the orthogonal complement of $\mu^\star$, and are optimal in the sense given by (7). In addition, diversity is enforced by requiring that the queries themselves be orthogonal to each other (Brinker 2003). Similar to the previous approach, the optimal solution is to pick the top k eigenvectors of the matrix $N^T \Sigma^\star\, N$, where N is the orthonormal basis to the orthogonal complement of $\mu^\star$. After that, we map those k eigenvectors into k synthetic queries using Eq. (9). The entire algorithm is summarized in Algorithm 1.

Evaluation

In our evaluations, we used the SOCP formulation in (6) because it is more computationally efficient. The optimization problem was solved using CVX (CVX Research 2012). For the alternative methods, training was done using linear SVM (Fan et al. 2008). Because the proposed algorithm targets the realizable noise-free setting, the experiments were carried out for various choices of problem dimension d and random choices of coefficient vectors $w \in \mathbb{R}^d$.

Single-Query Synthesis

In order to validate the proposed method in the single-query setting, we compare it against the following methods:

1. *Random Sampling*: Here, queries are chosen uniformly at random from the unit sphere.
2. *Uncertainty Sampling:* Here, queries are picked uniformly at random in the orthogonal complement of w,

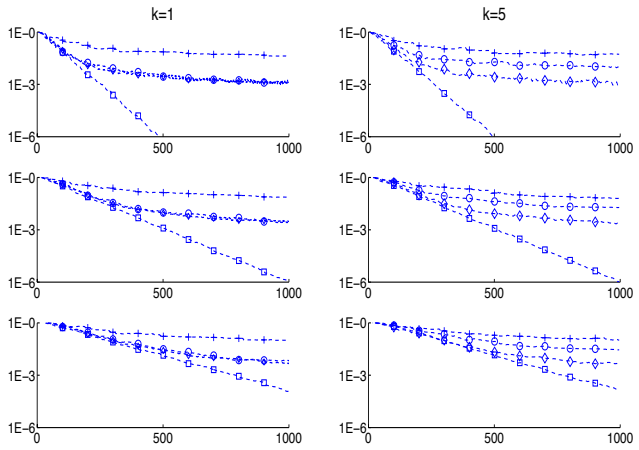

Figure 3: In this figure, the estimation error is plotted in a log-scale against the number of queries for various query synthesis algorithms. The left and right columns correspond to $k = 1$ (single-query) and $k = 5$ (batch-mode) respectively. The rows correspond to $d = 25, 50, 75$ from top to bottom respectively. The '+' curves are for *random sampling*. The '$\diamondsuit$' curves are for *uncertainty sampling* ($k = 1$, left) and *orthogonal sampling* ($k = 5$, right). The '$\circ$' curves are for the query-by-bagging method ($k = 1$, left) and the *representative sampling* method ($k = 5$, right). Finally, the '$\square$' curves are for the new proposed method.

where w is the coefficient vector learned using linear SVM. Because instances have zero margin, this method can be interpreted as the query synthesis analog to the pool-based uncertainty sampling method.

3. *Query-by-Bagging*: This method takes a *bagging* approach to implement the QBC algorithm (Abe and Mamitsuka 1998). Following (Abe and Mamitsuka 1998), we used a bag size of 20. At each iteration, a pool of 1,000 instances is randomly generated, and the query with the largest disagreement among the 20 hypotheses is picked.

Batch-Mode Query Synthesis

In the batch setting, where k queries are synthesized at each iteration, we compare the proposed algorithm against the following three methods:

1. *Random Sampling:* Here, the k queries are chosen uniformly at random from the unit sphere.
2. *Orthogonal Sampling:* In this method, k orthogonal queries are chosen uniformly at random from the orthogonal complement of w, where w is the coefficient vector learned using linear SVM. This can be interpreted as the query synthesis analog to the optimization-based method proposed in (Brinker 2003).
3. *Representative Sampling*: This is a clustering-based method proposed in (Xu et al. 2003). In our implementation, we used a pool size of 1,000 instances.

The exact experiment ran as follows. For a fixed choice of k and d, we began with a random choice of a unit-norm $w^\star \in \mathbb{R}^d$, a single positive example, and a single negative example. After that, we ran the different query synthesis algorithms in parallel up to a total of $1,000$ queries.

Data: Observations $\{(x_i, y_i)\}_{i=1,2,\ldots,t}$

Result: k synthetic queries $\{x_{t+1}, x_{t+2}, \ldots, x_{t+k}\}$

Begin:

1. Solve the optimization problem in (5) or (6). Let $\mu^\star$ and $\Sigma^\star$ be the optimal solutions.
2. Compute N, which is the orthonormal basis to the orthogonal complement of $\mu^\star$ (the null-space of $\mu^{\star T}$).
3. Compute $\alpha_1, \alpha_2, \ldots, \alpha_k$, which are the top k eigenvectors of the matrix $N^T \Sigma^\star N$.
4. Return $x_{t+1} = N\alpha_1, \ \ldots, \ x_{t+k} = N\alpha_k$.

Algorithm 1: Query synthesis algorithm for halfspaces.

At each iteration, we used past queries to estimate the true coefficient vector $w^\star$ using support vector machines (SVM) (Cortes and Vapnik 1995), which was implemented using the LIBLINEAR package (Fan et al. 2008). If we let $\hat{w}$ be the estimated coefficient vector, then estimation error is defined by $||w^\star - \hat{w}||_2$. Note that $||w^\star||_2 = ||\hat{w}||_2 = 1$ always holds, which implies that the estimation error is a proper measure of disagreement between the two halfspaces. In the batch setting, we used $k = 5$. Also, all experiments were repeated for $d \in \{25, 50, 75\}$. Figure 3 shows the evaluation results[1].

As shown in the figure, the proposed spectral algorithm *significantly* outperforms all other methods in both the single-query setting and the batch-mode setting. In fact, unlike the other methods whose estimation error is subject to *diminishing returns*, the new proposed method always maintains an exponential reduction in estimation error as indicated by the linear curve in this log-scale plot. In fact, the estimation error is approximately given by $\exp\{-\frac{m}{2d}\}$. This implies that in order to achieve an estimation error of $||w^\star - \hat{w}||_2 = \epsilon$, we only need $O(d \log \frac{1}{\epsilon})$ synthetic queries. As mentioned earlier, a sample complexity of $\Omega(d \log \frac{1}{\epsilon})$ can be shown to be optimal using a sphere counting argument (Dasgupta, Kalai, and Monteleoni 2009). Hence, the sample complexity of Algorithm 1 is near-optimal in practice (up to a leading constant).

Conclusions

In this paper, we propose a new query synthesis algorithm for learning halfspaces. Unlike pool-based sampling methods whose complexity grows exponentially fast with each iteration, complexity of the proposed algorithm grows only polynomially while still offering an exponential reduction in estimation error. Experimentally, the new algorithm significantly outperforms popular active learning strategies such as uncertainty sampling and representative sampling. It enjoys a sample complexity of $O(d \log \frac{1}{\epsilon})$ in practice, which can be shown to be near-optimal using a sphere counting argument.

[1]MATLAB implementation codes will be made available at http://mine.kaust.edu.sa/Pages/Software.aspx

References

Abe, N., and Mamitsuka, H. 1998. Query learning strategies using boosting and bagging. In *ICML*.

Angluin, D. 2001. Queries revisited. In *Algorithmic Learning Theory*, 12–31. Springer.

Balcan, M. F., and Long, P. M. 2013. Active and passive learning of linear separators under log-concave distributions. In *COLT*.

Balcan, M.-F.; Broder, A.; and Zhang, T. 2007. Margin based active learning. 35–50.

Brinker, K. 2003. Incorporating diversity in active learning with support vector machines. In *ICML*, 59–66.

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

Cortes, C., and Vapnik, V. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

CVX Research, I. 2012. CVX: Matlab software for disciplined convex programming, version 2.0. http://cvxr.com/cvx.

Dasgupta, S.; Kalai, A. T.; and Monteleoni, C. 2009. Analysis of perceptron-based active learning. *JMLR* 10:281–299.

Dasgupta, S. 2005a. Analysis of a greedy active learning strategy. *NIPS* 17:337–344.

Dasgupta, S. 2005b. Coarse sample complexity bounds for active learning. In *NIPS*.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. LIBLINEAR: A library for large linear classification. *JMLR* 9:1871–1874.

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

Haussler, D., and Warmuth, M. 1993. The probably approximately correct (PAC) and other learning models. 291–312.

King, R. D., et al. 2009. The automation of science. *Science* 324(5923):85–89.

Lewis, D. D., and Gale, W. A. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, 3–12. Springer-Verlag New York, Inc.

Liu, Y. 2004. Active learning with support vector machine applied to gene expression data for cancer classification. *Journal of chemical information and computer sciences* 44(6):1936–1941.

Lobo, M. S.; Vandenberghe, L.; Boyd, S.; and Lebret, H. 1998. Applications of second-order cone programming. *Linear algebra and its applications* 284(1):193–228.

McCallum, A., and Nigam, K. 1998. Employing EM and pool-based active learning for text classication. In *ICML*.

Mitchell, T. M. 1982. Generalization as search. *Artificial intelligence* 18(2):203–226.

Nelson, B.; Rubinstein, B. I.; Huang, L.; Joseph, A. D.; Lee, S. J.; Rao, S.; and Tygar, J. 2012. Query strategies for evading convex-inducing classifiers. *JMLR* 13:1293–1332.

Roy, N., and McCallum, A. 2001. Toward optimal active learning through monte carlo estimation of error reduction. In *ICML*.

Schein, A. I., and Ungar, L. H. 2007. Active learning for logistic regression: an evaluation. *Machine Learning* 68(3):235–265.

Sculley, D. 2007. Online active learning methods for fast label-efficient spam filtering. In *CEAS*.

Settles, B. 2010. Active learning literature survey. Computer Science Technical Report 1648, University of Wisconsin–Madison.

Settles, B. 2011. From theories to queries: Active learning in practice. *Challenges in Machine Learning* 6:1–18.

Tong, S., and Koller, D. 2002. Support vector machine active learning with applications to text classification. *JMLR* 2:45–66.

Valiant, L. G. 1984. A theory of the learnable. *Communications of the ACM* 27(11):1134–1142.

Vandenberghe, L.; Boyd, S.; and Wu, S.-P. 1998. Determinant maximization with linear matrix inequality constraints. *SIAM journal on matrix analysis and applications* 19(2):499–533.

Xu, Z.; Yu, K.; Tresp, V.; Xu, X.; and Wang, J. 2003. Representative sampling for text classification using support vector machines. In *Advances in Information Retrieval*, 393–407. Springer.

Yuan, G.-X.; Ho, C.-H.; and Lin, C.-J. 2012. Recent advances of large-scale linear classification. *Proceedings of the IEEE* 100(9):2584–2603.