A Version Space Perspective on Differentially Private Pool-Based Active Learning

Shantanu Rane and Alejandro E. Brito
Palo Alto Research Center (PARC)
Email: {srane,abrito}@parc.com

***Abstract*—We analyze pool-based active learning under a differential privacy guarantee. At every active learning step, some samples are selected to be labeled by an oracle, and the new labels are used to update the classifier. We want to preserve differential privacy during both the sample selection step and the classifier update step. To study the evolution of the active learner, we use the concept of a version space of possible hypotheses (classifiers). This concept helps establish a principled notion of the informativeness of a pool sample: When informative samples are labeled and used for training, the version space shrinks, yielding classifiers consistent with the labeled samples. To provide privacy, we query the oracle with both informative and non-informative samples using a simple randomized sampling scheme. We prove the privacy guarantee, and characterize the increase in label complexity resulting from our randomized sampling strategy. To examine how our theoretical analysis manifests in practice, we built an SVM-based active learner, and measured the accuracy and label complexity achieved with and without privacy.**

I. Introduction

Traditional non-interactive supervised learning is often label-hungry, i.e., a very large number of labeled samples are used to train an accurate classifier. In contrast, active learning approaches seek to train a classifier using fewer *informative* samples. This is particularly useful when very little labeled data is available, or when labeling is expensive. We are interested in privacy-aware variants of the active learning workflow, which are relevant in many practical applications. One example is federated learning, in which an accurate model is to be trained using data that is distributed over a large number of clients [1]. To achieve this, the centralized node sends an initial, coarse model to the clients, asks each client to independently update the model based on its local data, and then aggregates the clients' individual models. Though the clients do not send any data to the aggregator, this approach does *not* guarantee privacy. In fact, an adversarial aggregator can observe the client models and make inferences about a client's sensitive data. Differentially private mechanisms were proposed to provide strong statistical guarantees against the success of such attacks. Recently, researchers have developed differentially private federated learning mechanisms in the non-interactive supervised learning paradigm where an abundance of labeled data is available. Development and analysis of privacy-aware mechanisms for active learning is the primary focus of this paper.

II. Related Work and Contributions

Differential privacy has been extensively studied and applied since its introduction in 2006 [2]; Dwork and Roth's book [3] provides an inclusive review. The literature on differentially private mechanisms for active learning is relatively sparse. Ghassemi *et al.* [4] trained a differentially private anomaly detector using active learning in a streaming (online) modality. They selected informative samples from the data stream for labeling by an oracle, and updated a classifier using the new labels. To identify and select informative samples, the authors modified a heuristic developed by Tong and Koller [5] and adapted it to the differential privacy setting. Notably, they incorporated differential privacy both in the sample selection step, as well as in the classifier update step. We adopt this general privacy approach as well, though for the pool-based rather than the online setting. More importantly, our implementation differs in one important aspect: a data sample that is not initially chosen for labeling can be returned to the pool for a possible later labeling opportunity.

A key contribution of this paper is the analysis of the differentially private mechanism using the version space concept [6]. This concept has been used in the analysis of active learning, in particular, to prove convergence of the classifier and to derive bounds on the label complexity [7]–[9]. Indeed, the Tong-Koller heuristics for choosing informative samples are also based on a version space analysis [5], albeit privacy was not a consideration there. Analyzing the evolution of a differentially private active learner from the perspective of its steadily shrinking version space is useful in at least two respects: (1) It provides an principled approach to choose samples for labeling while preserving privacy, and (2) It indicates when adding noise to a classifier for the purpose of achieving differential privacy can also make it less accurate. This, in turn, reveals good and bad ways to perform the differentially private classifier update step.

This paper is organized as follows: In Section III, we introduce the concept of a version space and use it to analyze an active learning workflow without privacy considerations. In Section IV, we describe the differentially private mechanism, prove the privacy guarantee, and analyze the increase in label complexity due to the privacy mechanism. We describe an implemention of a SVM-based active learner in Section V. Using this learner, we discuss how some of the theoretical ideas are manifested in the experiments described in Section VI.

III. Active Learning Setting: No privacy

We briefly review the concept of the version space and the disagreement coefficient associated with a classifier. Throughout this paper, we will consider only the 2-class problem, as widely encountered in concept learning, anomaly detection, and other related problems. For simplicity, we restrict the development to the case in which the two classes are linearly separable, though generalizing to the agnostic case is also

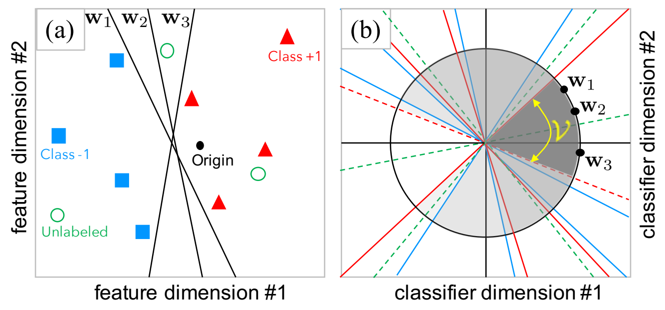


Fig. 1. (a) Typical active learning setting with a few labeled samples and a large pool of unlabeled samples. 3 classifiers, $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$ in the current version space are shown. (b) Dual representation in which classifiers appear as points while data samples appear as hyperplanes (lines in 2D). The version space $\mathcal{V}$ is the set of points on the unit circle satisfying $y_i h(\mathbf{x}_i) = y_i(\mathbf{w}^T \mathbf{x}_i) > 0$ which is the intersection of half-circles determined by lines representing the training data samples. The dashed green hyperplane corresponding to one green unlabeled sample intersects $\mathcal{V}$. This sample is informative and a good candidate for querying. The other green unlabeled samples are represented by hyperplanes that pass outside $\mathcal{V}$, and are thus considered non-informative.

possible. Thus, the classifier is a hyperplane that separates the data into classes with labels ± 1. Suppose that the active learner has n labeled training samples denoted by the (sample, label) pairs as $\mathcal{T} = \{(\mathbf{x}_i, y_i), \mathbf{x}_i \in \mathbb{R}^d, i = 1, 2, \cdots, n\}$; a pool of m unlabeled samples, $\mathcal{M} = \{\mathbf{z}_j \in \mathbb{R}^d, j = 1, 2, \cdots, m\}$. The $\mathbf{x}_i$ and $\mathbf{z}_j$ above belong to an input space $\mathcal{X}$. An initial classifier $\mathbf{w}_0$ has been trained on $\mathcal{T}$. We will ask an oracle for labels of *a few* samples from $\mathcal{M}$, and train the learner to be consistent with all available labels. The rationale is that such a learner will accurately classify data whose distribution matches that of the pool.

A. Version Space

For labeled data $\mathbf{x}_i, i = 1, 2, \cdots, n$ separated by a hyperplane $\mathbf{w}$, we define a hypothesis $h(\cdot)$ as $h(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i / \|\mathbf{w}\|$ where $\|\cdot\|$ is the ℓ_2 norm and $(\cdot)^T$ is the transpose operator. Then, in the separable case, a label is assigned as $y_i = 1$ if $h(\mathbf{x}_i) > 0$ and $y_i = -1$ otherwise. Thus, $y_i h(\mathbf{x}_i) > 0$.

Definition 1 *The version space $\mathcal{V}$ is the set of all possible hypotheses that separate the labeled data in the feature space $\mathcal{X}$ [6]. We define the version space in terms of the hypotheses h as well as the hyperplanes $\mathbf{w}$ as*

$$\begin{aligned}\mathcal{V} &= \{h \in \mathcal{H} \,|\, \forall i \in \{1, \cdots, n\},\, y_i h(\mathbf{x}_i) > 0\}. \\ &= \{\mathbf{w} \in \mathcal{W} \,|\, \|\mathbf{w}\| = 1, y_i(\mathbf{w}^T \mathbf{x}_i) > 0, i = 1, \cdots, n\}\end{aligned}$$

We will use the version space concept to describe the evolution of the active learner, both in the non-private and differentially private case. For this purpose (See Fig. 1), consider a dual representation in which the points in the input space $\mathcal{X}$ are hyperplanes in the hypothesis space $\mathcal{W}$, while candidate separators $\mathbf{w}$ are just points in $\mathcal{W}$. In this representation, it can be shown that the optimal classifier $\mathbf{w}^*$ is the center of mass of $\mathcal{V}$. An approximation to $\mathbf{w}^*$ is a classifier that maximizes the margin with respect to each class, given by a Support Vector Machine (SVM) classifier [5].

B. Active Learning in the Pool-based Setting

For the non-private case, we consider the popular CAL algorithm for active learning in the separable case [10]. The approach we describe may not necessarily be constructive, and is meant to develop a theoretical understanding. To construct an actual active learner for our experiments, we make some modifications described in Section V. The active learner's task is to query an oracle for labels of points in $\mathcal{M}$, and using the received labels, to keep updating both the version space and the classifier. Let $t = 1, 2, ..., T$ denote the step number at which the classifier and version space are updated. Let $\mathcal{L}_t$ be the set of samples that have been queried, labeled and removed from the pool $\mathcal{M}$ after the end of the t^{th} step. Define $\mathcal{L}_0 = \Phi$, the empty set. At the beginning of the (t^{th}) step, suppose that c unlabeled samples are drawn from $\mathcal{M} \backslash \mathcal{L}_{t-1}$, where $\backslash$ is the set difference operator. After training using the newly available labels, we release the classifier $\mathbf{w}_t$.

C. Informative Samples Reduce the Version Space

Denote the version space after the t^{th} step by $\mathcal{V}_t$. Recall that the points in the current pool, $\mathcal{M} \backslash \mathcal{L}_t$, belong to the input space $\mathcal{X}$, and are thus hyperplanes in the hypothesis space $\mathcal{W}$. By definition (See Fig 1(b) for intuition), hyperplanes that do not intersect $\mathcal{V}_t$ do not provide useful information in terms of improving the learner's predictive capability, and they need not be queried. By contrast, a hyperplane corresponding to a sample that intersects $\mathcal{V}_t$ indicates that some classifiers in $\mathcal{V}_t$ must be removed because they would classify that sample incorrectly. Therefore, we query the labels of such informative pool samples and obtain a new smaller version space $\mathcal{V}_{t+1} \subset \mathcal{V}_t$. With the new labels, a classifier $\mathbf{w}_{t+1}$ is trained which, by construction, is consistent with all the correct predictions that $\mathbf{w}_t$ could make. This process is repeated until we obtain the version space $\mathcal{V}_T$ and the classifier $\mathbf{w}_T$.

D. Label Complexity

Label complexity is defined as the number of labels that must be obtained before the classifier can be trained to a desired accuracy. For traditional non-interactive supervised learning, the label complexity required to reach an error probability $\eta \in (0, 1]$ with respect to the optimal classifier is given by $\Omega(1/\eta)$ [11]. In practice, the accuracy or error probability is computed over a labeled dataset – termed the "holdout" dataset – that is representative of the underlying data distribution but is not used in training. Since invoking the oracle in active learning is costly, we must control the label complexity. It is well known that applying *active* learning heuristics to choose only informative samples to be queried, incurs significantly lower label complexity than non-interactive supervised learning which trains on all samples.

Lemma 1 *[12] The active learning workflow described in Section III-C will output a hypothesis with error less than η with high probability, after $O(\log(1/\eta))$ rounds.*

The reader is referred to [12] for the proof. The proof requires that a large enough number of *informative* samples (denoted by $\gamma \leq c$) be labeled prior to learning $\mathbf{w}_t$, for every t. γ depends on the Vapnik-Chervonenkis (VC) dimension of the classifier and the "disagreement coefficient", both of which are defined and elaborated in [12]. For our purposes, it is sufficient to note that choosing γ samples ensures that the version space $\mathcal{V}_t$ shrinks fast enough with increasing t, resulting in more and more accurate classifiers. The label complexity of active learning is then given by $O(\gamma \log(1/\eta))$. In other words, the label complexity is proportional to $\log(1/\eta)$, compared to $1/\eta$ for non-interactive supervised learning.

IV. Active Learning with Differential Privacy

Let us analyze pool-based active learning under the differential privacy paradigm. We describe our adversarial model, and then give the privacy-aware active learning workflow. We show that this workflow satisfies the differential privacy guarantees. Then, we quantify the price paid for privacy in the form of increased label complexity. We examine the effects of differentially private mechanisms in the version space. Our study suggests that careful consideration of the privacy/performance tradeoff is necessary while updating the active learner in the differentially private setting.

A. Adversarial Model under Differential Privacy

The adversarial model is slightly different from the one usually encountered for the standard supervised learning scenario. Since the learner typically starts with a small training set (i.e., n is small), we don't seek to protect the privacy of the training set. We seek instead to protect the privacy of the pooled samples whose labels are queried and used to update the classifier. Hence we assume that the adversary knows the training set of n samples. Further, the adversary possesses an *adjacent* pool of m samples denoted by $\mathcal{M}' = \{\mathbf{z}'_j, j = 1, 2, \cdots, m\}$, where there is a particular $i \leq m$ such that $\mathbf{z}'_i \neq \mathbf{z}_i$ while for all valid $j \neq i$ it holds that $\mathbf{z}'_j = \mathbf{z}_j$. Crucially, the adversary does not know the index i of the sample that differs.

We require that the adversary should not be able to identify $\mathbf{z}_i$ by observing any model update $\mathbf{w}_t$. Furthermore, he should not be able to find out whether $\mathbf{z}_i$ was used to train $\mathbf{w}_t$. Let the vector of classifiers be $\mathbf{W}_T = (\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_T)$. Let c unlabeled samples be drawn from the pool and examined at each step, as in the non-private case. Let $\mathbf{Q}_T = (\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_T)$ with $\mathbf{S}_k = (s_{(k-1)c+1}, ..., s_{kc})$. Here, $\mathbf{S}_k, k \in \{1, \cdots, T\}$ is a binary vector of length c, containing selection results, in which $s_{(k-1)c+j} = 1$ if the jth sample from $\mathbf{S}_k$ was chosen for labeling by the oracle, and $s_{(k-1)c+j} = 0$ if it was not chosen. We require that the probability of deriving a certain classifier will change by at most a small multiplicative factor, even if the differing sample was used for training the learner. Concretely, using the definition of ϵ-differential privacy [3]:

$$P(\mathbf{W}_T, \mathbf{Q}_T | \mathcal{M}) \leq \exp(\epsilon) P(\mathbf{W}_T, \mathbf{Q}_T | \mathcal{M}')$$

We remark that the adversary's view includes only the adjacent pool $\mathcal{M}'$, and the outputs of the adversary are the vector of classifiers $\mathbf{W}_T$ and the binary vector $\mathbf{Q}_T$ that indicates which samples are chosen for labeling.

B. Differentially Private Active Learning Workflow

As before, an initial classifier $\mathbf{w}_0$ has been trained on n training samples. By our assumptions, $\mathbf{w}_0$ is available to the adversary. We want to improve the classifier's accuracy with the help of the pool $\mathcal{M}$, without revealing to the adversary which samples in $\mathcal{M}$ caused the model updates. We retain the same publishing schedule, i.e., c samples are queried at each step, the classifier models $\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_T$, are published and available to the adversary. To achieve privacy, we propose the following privacy-aware version of the workflow in Section III.

Suppose that, we are at the *beginning* of the tth step of the model update process, the version space is $\mathcal{V}_{t-1}$, the corresponding model maintained by the learner is $\mathbf{w}_{t-1}$. The learner has access to the pool $\mathcal{U}_{t-1} = \mathcal{M} \backslash \mathcal{L}_{t-1}$. Now, consider the hyperplanes in the hypothesis space $\mathcal{W}$ corresponding to the unlabeled samples in $\mathcal{U}_{t-1}$. Some of these hyperplanes intersect $\mathcal{V}_{t-1}$, while others pass outside it.

Recall that a hyperplane that intersects $\mathcal{V}_{t-1}$ represents an informative sample whose label should be queried. However, in the differentially private workflow, we must adopt a different approach to choose samples for querying. Concretely, for each $i = 1, 2, \cdots, |\mathcal{U}_{t-1}|$, if the hyperplane corresponding to $\mathbf{z}_i \in \mathcal{U}_{t-1}$ passes through $\mathcal{V}_{t-1}$, query $\mathbf{z}_i$ with probability $p > 1/2$. Otherwise, if the hyperplane corresponding to $\mathbf{z}_i$ passes outside $\mathcal{V}_{t-1}$, query $\mathbf{z}_i$ with probability $1 - p$. If $\mathbf{z}_i$ was informative but not chosen for querying, it is returned to the pool for possible querying later. If $\mathbf{z}_i$ was non-informative and not chosen for querying, it is discarded from the pool. The procedure is repeated for $\mathbf{z}_{i+1}$ until c samples from $\mathcal{U}_{t-1}$ have been examined. This is inefficient compared to the non-private version because not all informative samples (those whose hyperplanes intersect $\mathcal{V}_{t-1}$) are chosen, and some non-informative samples (those whose hyperplanes do not intersect $\mathcal{V}_{t-1}$) are chosen. The inefficiency depends on p.

Denote the non-private classifier trained using the newly labeled points by $\mathbf{w}^\dagger_t$. From this classifier, we derive and release a ϵ_m-differentially private classifier $\mathbf{w}_t$. We emphasize that, at each update step, the adversary's view includes the previous released (differentially private) classifier $\mathbf{w}_{t-1}$, an adjacent pool $\mathcal{M}'$ and the binary vector $\mathbf{S}_t$, indicating which samples have been chosen for labeling just before updating the classifier, as defined in Section IV-A. Thus, applying the definition of differential privacy, we have for each $t < T$,

$$P(\mathbf{w}_t | \mathbf{w}_{t-1}, \mathbf{S}_t, \mathcal{M}) \leq \exp(\epsilon_m) P(\mathbf{w}_t | \mathbf{w}_{t-1}, \mathbf{S}_t, \mathcal{M}') \quad (1)$$

Our approach is agnostic to the particular mechanism used to achieve ϵ_m-differential privacy in $\mathbf{w}_t$. Thus, output perturbation, or objective perturbation or the exponential mechanism could be used. To reiterate, (a) the model updates $\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_T$ are derived using a differentially private mechanism, and (b) $\mathbf{w}^\dagger_T$ has the desired accuracy η. Now, we can make a few privacy claims.

Proposition 1 *As described above, at each step t, let Bernoulli(p) sampling be used to query samples whose hyperplanes intersect $\mathcal{V}_{t-1}$, and Bernoulli($1-p$) sampling is used to query samples whose hyperplanes do not intersect $\mathcal{V}_{t-1}$. For $p \geq 1/2$, this selection procedure is ϵ_p-differentially private with $\epsilon_p = \log \frac{p}{1-p}$.*

Proof: Assume that the samples in $\mathcal{M}$ and $\mathcal{M}'$ are ordered consistently. This is a conservative assumption, but the situation can occur, for example, if the learner and the adversary use a known algorithm to rank unlabeled samples based on their informativeness. The adversary has observed $\mathbf{w}_t$ and knows its version space $\mathcal{V}'_t$. Since $\mathcal{M}$ and $\mathcal{M}'$ differ in one element, $\mathcal{V}'_t$ may or may not be the same as $\mathcal{V}_t$.

Let s_i denote the selection variable for a pool sample $\mathbf{z}_i \in \mathcal{M}$ or $\mathbf{z}'_i \in \mathcal{M}'$. $s_i = 1$ indicates that $\mathbf{z}_i$ (or equivalently $\mathbf{z}'_i$) is selected for querying, while $s_i = 0$ indicates otherwise. Then, to construct the bound for $s_i = 1$, we use the worst case situation in which the hyperplane corresponding to $\mathbf{z}_i$ intersects $\mathcal{V}_t$ but the hyperplane corresponding to $\mathbf{z}'_i$ does not intersect $\mathcal{V}'_t$. Thus,

$$\left|\log \frac{P(s_i = 1|\mathcal{M}, \mathbf{w}_t)}{P(s_i = 1|\mathcal{M}', \mathbf{w}_t)}\right|$$
$$< \left|\log \frac{P(s_i = 1|\mathbf{z}_i \text{ intersects } \mathcal{V}_t)}{P(s_i = 1|\mathbf{z}'_i \text{ does not intersect } \mathcal{V}'_t)}\right| = \left|\log \frac{p}{1-p}\right|$$

The case $s_i = 0$ is argued similarly, for the worst case where the hyperplane corresponding to $\mathbf{z}_i$ does not intersect $\mathcal{V}_t$ but the hyperplane corresponding to $\mathbf{z}'_i$ intersects $\mathcal{V}'_t$. As $p \geq 1/2$, we drop the absolute value notation and the result follows. ■

Theorem 1 *Suppose the differentially private learner is trained over T steps with c samples labeled per step. Let $|\mathcal{M}| = |\mathcal{M}'| > Tc$. The released classifiers are ϵ-differentially private, where $\epsilon = \epsilon_m + \epsilon_p$.*

Proof: Again assume that the adjacent pools $\mathcal{M}$ and $\mathcal{M}'$ are indexed consistently. Without loss of generality, set $\mathbf{W}_0 = 0$. The classifier is updated at each step using a ϵ_m-differentially private mechanism. Then, we have,

$$\frac{P(\mathbf{W}_T, \mathbf{Q}_T|\mathcal{M})}{P(\mathbf{W}_T, \mathbf{Q}_T|\mathcal{M}')} = \prod_{t=1}^{T} \frac{P(\mathbf{w}_t, \mathbf{S}_t|\mathcal{M}, \mathbf{W}_{t-1})}{P(\mathbf{w}_t, \mathbf{S}_t|\mathcal{M}', \mathbf{W}_{t-1})}$$
$$= \prod_{t=1}^{T} \frac{P(\mathbf{w}_t|\mathbf{w}_{t-1}, \mathcal{M}, \mathbf{S}_t)P(\mathbf{S}_t|\mathbf{w}_{t-1}, \mathcal{M})}{P(\mathbf{w}_t|\mathbf{w}_{t-1}, \mathcal{M}', \mathbf{S}_t)P(\mathbf{S}_t|\mathbf{w}_{t-1}, \mathcal{M}')} \tag{2}$$
$$= \frac{P(\mathbf{w}_i|\mathbf{w}_{i-1}, \mathcal{M}, \mathbf{S}_i)}{P(\mathbf{w}_i|\mathbf{w}_{i-1}, \mathcal{M}', \mathbf{S}_i)} \prod_{k\in\tau} \frac{P(\mathbf{S}_k|\mathbf{w}_{k-1}, \mathcal{M})}{P(\mathbf{S}_k|\mathbf{w}_{k-1}, \mathcal{M}')} \tag{3}$$
$$= \frac{P(\mathbf{w}_i|\mathbf{w}_{i-1}, \mathcal{M}, \mathbf{S}_i)}{P(\mathbf{w}_i|\mathbf{w}_{i-1}, \mathcal{M}', \mathbf{S}_i)} \prod_{k\in\tau} \prod_{1\leq j\leq c} \frac{P(s_{(k-1)j+c}|\mathbf{w}_{k-1}, \mathcal{M})}{P(s_{(k-1)j+c}|\mathbf{w}_{k-1}, \mathcal{M}')} \tag{4}$$

where the sample that differs between $\mathcal{M}$ and $\mathcal{M}'$ was chosen for querying at step $i \leq T$. To obtain (2), observe that $\mathbf{w}_t$ depends only on samples from $\mathcal{M}$ and on the immediately previous classifier $\mathbf{w}_{t-1}$. Note that step i is not necessarily the first step the differing sample was encountered. For instance, the sample could have been informative at step $t < i$, i.e., intersecting the version space $\mathcal{V}_{t-1}$ but was not selected for querying in the Bernoulli sampling process, and thus returned to the pool. This creates the possibility of the differing sample being chosen at a later step. The set $\tau \subseteq \{1, 2, \cdots, T\}$ in the second term of (3) is the set of all steps at which the differing sample could be encountered in our pool-based active learning scenario. This is different from a stream-based (online) settings in which the differing sample is seen only once, whether it is chosen for querying or not [4]. To obtain (4), we use the definition of $\mathbf{S}_k$ in Section IV-A.

For $p \geq 1/2$, the double product term in (4) is maximized in two cases: Either (a) the differing sample, which belongs to $\mathcal{M}$, intersects $\mathcal{V}_{i-1}$ and is queried by the learner, whereas its counterpart in $\mathcal{M}'$ does not intersect $\mathcal{V}'_{i-1}$ but is queried by the adversary or (b) At any step $t \leq T$, the differing sample does not intersect $\mathcal{V}_{t-1}$ and is not queried by the learner, whereas its counterpart in $\mathcal{M}'$ intersects $\mathcal{V}'_{t-1}$ but is not queried by the adversary. The probability ratio is $p/(1-p)$ in either case. Since we stipulated above that a non-informative sample – one whose hyperplane does not intersect $\mathcal{V}_{t-1}$ – that is not chosen for querying is removed from the pool, the situation (b) occurs at most once in T steps. So, we can bound the ratio in (3) as:

$$\left|\log \frac{P(\mathbf{W}_T, \mathbf{Q}_T|\mathcal{M})}{P(\mathbf{W}_T, \mathbf{Q}_T|\mathcal{M}')}\right|$$
$$\leq \left|\log \frac{P(\mathbf{w}_i|\mathbf{w}_{i-1}, \mathcal{M}, \mathbf{S}_i)}{P(\mathbf{w}_i|\mathbf{w}_{i-1}, \mathcal{M}', \mathbf{S}_i)}\right| + \left|\log \frac{p}{1-p}\right| \leq \epsilon_m + \epsilon_p.$$

where the last inequality follows from Proposition 1 and the definition of ϵ_m in (1). ■

C. Effect of Privacy on Label Complexity

Proposition 2 *For $1/2 \leq p < 1$ and classification error probability η, consider the privacy-aware active learning workflow described in Section IV-B. The label complexity of this approach is $O((1/p)\log(1/\eta))$.*

Proof: As we noted earlier (Lemma 1), the active learning algorithm without privacy outputs a hypothesis with error probability less than η in $O(\log(1/\eta))$ rounds, provided at least γ informative samples are labeled at the t^{th} step. In contrast, for the differentially private case, samples whose hyperplanes intersect $\mathcal{V}_t$ are queried only with probability p. Other *non-informative* samples are queried with probability $1-p$ to create uncertainty about which samples from the pool are labeled. The non-informative samples do not contribute to the shrinkage of $\mathcal{V}_t$. Thus, to ensure that at least γ informative samples are labeled per privacy-preserving selection step, it is necessary to query $O(\gamma/p)$ samples per step. With this larger number of per-step queries, the conditions of Lemma 1 are again met and we obtain a hypothesis with error less than η with high probability, after $O(\log(1/\eta))$ rounds. The effective label complexity is thus $O((\gamma/p)\log(1/\eta))$, and the result follows. As $p \geq 1/2$, this is only a moderate increase in label complexity. ■

D. Version Space View of Utility of DP mechanisms

The differentially private classifier $\mathbf{w}_t$ is, at best, a noisy approximation of the optimal classifier $\mathbf{w}^*_t$ corresponding to the version space $\mathcal{V}_t$. For the linearly separable case, if the noise is small enough to keep $\mathbf{w}_t$ inside $\mathcal{V}_t$, its consistency with respect to $\mathcal{V}_t$ is preserved. However, if too much noise is added, then $\mathbf{w}_t$ moves out of $\mathcal{V}_t$, which means that it will classify some of the labeled samples incorrectly.

This has important implications for how we evolve the classifier $\mathbf{w}_t$. One way is to update the previous noisy classifier $\mathbf{w}_{t-1}$ using the new labels obtained in the t^{th} step. The preceding argument, however, suggests that this might compromise the classifier's consistency with respect to the version space. A better approach is to (a) preserve all the samples labeled by the oracle until step t, and then (b) train a differentially private $\mathbf{w}_t$ from those samples, without using $\mathbf{w}_{t-1}$.

V. An SVM-based Active Learner

Consider an experimental learner designed to evaluate SVM-based active learning with and without privacy. For simplicity, only one sample is queried at each step t and its label is added to the set of already known labels. Using the available labels, a new non-private classifier $\mathbf{w}_t^{\text{SVM}}$ is trained using a dual SVM solver [13]. To choose the most informative sample for labeling in the non-private case, we resort to the following heuristic approach, called uncertainty sampling, developed by Tong and Koller [5]: Choose the sample closest to the hyperplane representing the SVM classifier. This gives a concrete approach to training the active learner without having to explicitly maintain the version space. To see why this is reasonable, recall that the optimal classifier $\mathbf{w}_t^*$ is the center of mass of the version space $\mathcal{V}_t$. Freund showed that choosing the pool sample $\mathbf{z}$ whose hyperplane halves $\mathcal{V}_t$ reduces the error of $\mathbf{w}_t^*$ exponentially with the number of queried samples [14]. Then, if $\mathcal{V}_t$ has a regular shape, the hyperplane corresponding to $\mathbf{z}$ would pass very close to $\mathbf{w}_t^*$. Moreover, it turns out that $\mathbf{w}_t^{\text{SVM}}$ is an approximation to $\mathbf{w}_t^*$ [5]. Hence, we choose to query the sample $\mathbf{z}$ whose hyperplane is closest to $\mathbf{w}_t^{\text{SVM}}$. This heuristic approach leverages the version space-based development from the previous section, without requiring us to explicitly keep track of $\mathcal{V}_t$. Concretely, this way of choosing a sample $\mathbf{z}$ to be queried ensures that $\mathcal{V}_t$ keeps shrinking reasonably fast with increasing t. As a consequence, a sequence of increasingly accurate classifiers, $\mathbf{w}_t^{\text{SVM}}$ are learned. In the non-private case, each released classifier is given by $\mathbf{w}_t = \mathbf{w}_t^{\text{SVM}}$.

To privately select a sample for querying, we maintain a ranked list of pool points, based on their distance to $\mathbf{w}_{t-1}^{\text{SVM}}$. Then, we implement a Bernoulli-p trial, i.e., toss coin with bias p and if the coin lands heads, query the top-ranked, i.e., closest pool point. If the coin lands tails, repeat the Bernoulli-p trial with the second closest pool point, and so on, until a sample is queried. All samples not chosen for querying are returned to the pool for ranking and possible re-use in subsequent learning steps. We then retrieve the label of the single queried sample, add it to the set of already known labels, and use the dual SVM solver to derive a new clean classifier $\mathbf{w}_t^{\text{SVM}}$. To guarantee differential privacy in the update step, we use the sensitivity-based output perturbation approach [3], where scalar zero-mean Laplace-distributed noise is added to each component of $\mathbf{w}_t^{\text{SVM}}$. Thus, each released private classifier is given by $\mathbf{w}_t = \mathbf{w}_t^{\text{SVM}} + \boldsymbol{\nu}_t$. To obtain $\boldsymbol{\nu}_t$, we follow the approach developed by Rubinstein *et al.* for non-interactive supervised SVM learning [15]. In particular, the scale parameter λ_t of the Laplacian noise components $\nu_t^i, i = 1, \cdots, d$ is given by:

$$\lambda_t \geq \frac{4LC\kappa\sqrt{d}}{\epsilon_m n_t} \tag{5}$$

where L is the Lipschitz constant, C is a cost parameter in the SVM dual optimization problem, κ is the kernel upper bound, d is the feature dimension and n_t is the number of labeled samples in $\mathcal{L}_t \cup \mathcal{T}$ used to train the active learner at step t. With the default ℓ_2-norm kernel, we have $L = 1, \kappa = 1$ and C is an input to the dual SVM solver. The distribution of the noise added to the i^{th} component of $\mathbf{w}_t^{\text{SVM}}$ is then given by:

$$f(\nu_t^i) = \frac{1}{2\lambda_t} \exp\left(-\frac{|\nu_t^i|}{\lambda_t}\right) = \text{Laplace}(0, \lambda_t) \text{ for } i = 1, ..., d$$

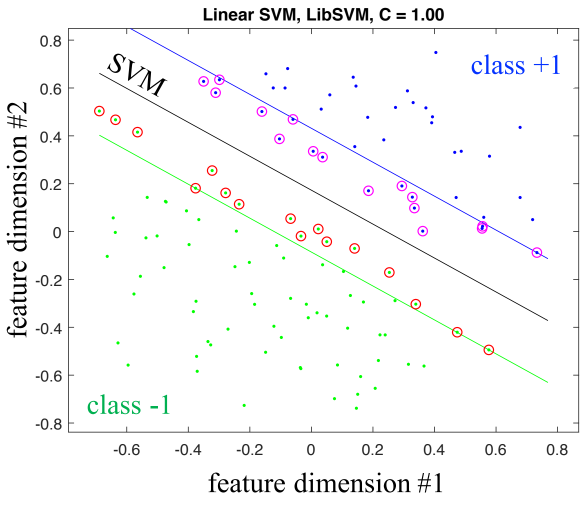


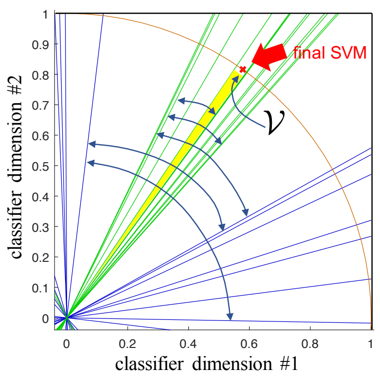
Fig. 2. The decision boundary of the non-private SVM is shown in black, with circled points being the support vectors. The blue and green lines represent the margins $\pm$ 1. This dataset is linearly separable using $C = 1$ in (5).

The inverse dependence of λ_t on n_t indicates a second privacy-utility tradeoff in addition to the increase in label complexity: Although active learning guarantees that $n_t \ll |\mathcal{M}|$, the inverse dependence unfortunately means that a classifier trained on n_t samples should be released with more noise than one trained on all $|\mathcal{M}|$ samples. The extra noise may shift $\mathbf{w}_t$ out of $\mathcal{V}_t$, the version space of the corresponding noiseless classifier, thereby reducing its accuracy.

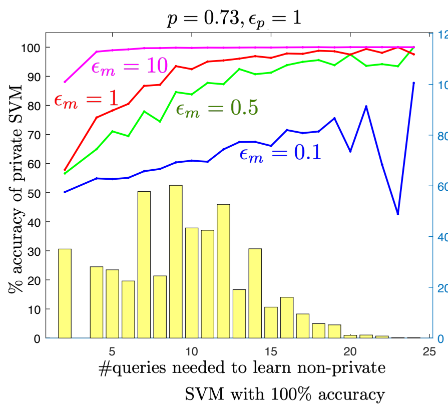
VI. Experimental Evaluation

We generated a synthetic dataset of 120 2-dimensional points in two linearly separable classes. The constellation of points, and the SVM decision boundary in the non-private case are shown in Fig. 2. In each experimental run, the SVM-based active learner is seeded with two randomly selected training samples, one from each class. The remaining points are assigned to the pool $\mathcal{M}$. Fig. 3(a) depicts the version space and the final, learned non-private classifier in one experimental run. The version space plot uses the fact that each point $\mathbf{x}_i$, label y_i, and consistent classifier $\mathbf{w}$ satisfy $y_i(\mathbf{w}^T\mathbf{x}_i) > 0$. Substituting the values of $\mathbf{x}_i, y_i$ in the inequality, we obtain intervals for consistent classifiers on the unit circle.

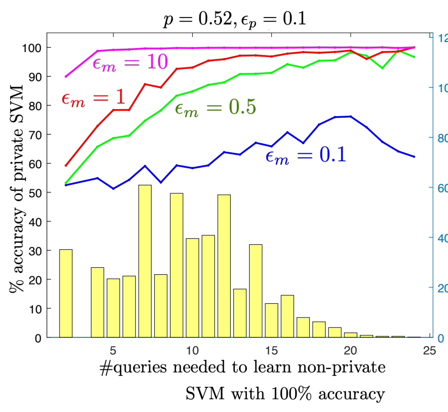
Our objective is to examine (1) the effect of differential privacy in the selection step (ϵ_p) on the label complexity, and (2) the effect of differential privacy in the update step (ϵ_m) on the accuracy of the final released classifier $\mathbf{w}_T$. For each privacy setting, i.e., (ϵ_p, ϵ_m), we run the differentially private active learning experiment 5000 times. The label complexity histograms in Figs. 3(b) and (c) are obtained for ϵ_p values of 0.1 and 1 corresponding to $p = 0.52$ and 0.73 respectively. Since label complexity is measured with respect to the sequence of clean (un-released) non-private classifiers, it is impacted only by ϵ_p and not by ϵ_m. The empirical label complexity is reported as the number of queries needed until the non-private SVM achieves 100% accurate separation of the data into classes +1 and -1. The privacy mechanism of randomizing the selection of the samples to be labeled does not appear to adversely affect label complexity in our experiments.



(a)



(b)



(c)

Fig. 3. (a) The 2-dimensional version space corresponding to this dataset. Blue and green lines represent the class +1 samples and class -1 samples respectively, and the red cross represents the final, trained, non-private SVM. The arcs show how the version space progressively shrinks as more and more informative points are labeled, until it is a tiny arc of the (red) unit circle. (b,c) The label complexity is *not significantly affected* by a change in ϵ_p. The accuracy of the final released differentially private classifier $\mathbf{w}_T$ is adversely affected if the classifier update is made more private (i.e., ϵ_m is lowered), or if the active learner had used very few labels. This is a direct consequence of the inverse relationship between the scale parameter λ_t and the number of labeled samples n_t in (5).

The rate at which the randomized selection shrinks the version space is faster than choosing the closest sample to $\mathbf{w}_t^{\text{SVM}}$ in some experimental runs, and slower in other runs. Thus, label complexity is sometimes slightly lower, and sometimes slightly higher than that observed for the non-private case.

The accuracy plots in Figs. 3(b) and (c) are obtained for $\epsilon_m \in \{0.1, 0.5, 1, 10\}$. Suppose that a empirical label complexity of ℓ is observed in $N(\ell)$ experimental runs. We compute the accuracy of the private SVM averaged over $N(\ell)$ runs, for each ℓ, and report those values in the plots of Figs. 3(b) and (c). With more noise in the classifier update, i.e., with smaller ϵ_m, the accuracy falls below 100%. Additionally, Figs. 3(b) and (c) reveal a kind of "no free lunch" tradeoff: Learners that use more labeled samples are more accurate. This is because they need less noise for the same amount of privacy. Conversely, learners that use fewer labeled samples result in more error-prone privacy-aware classifiers because they need more noise for the same amount of privacy.

VII. Conclusions

We studied differentially private active learning from the perspective of its steadily shrinking version space, and proved the privacy guarantees. Our analysis reveals tradeoffs that must be considered in the design of differentially private active learning schemes. Firstly, privacy-aware sample selection causes only a moderate increase in the label complexity. Secondly, privacy-aware learner update requires adding noise to the classifier, which might reduce its accuracy. The amount of noise added can be *significantly more* than that observed in non-interactive supervised learning because fewer samples are used for training the active learner. Thus, if a non-private active learner achieves the desired accuracy too fast, the accuracy of the corresponding released private classifier is compromised. Furthermore, care must be taken to ensure that noise added in successive update steps does not have a cummulative detrimental effect on the classifier's accuracy. Therefore, it is preferable to train the active learner anew at each querying step, using all *available labeled samples*, rather than updating an existing noisy learner.

Acknowledgments

We thank the anonymous reviewers and our shepherd, Dr. Juan Ramón Troncoso-Pastoriza, for numerous suggestions that helped us clarify the content and presentation of this work.

References

[1] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers. Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*, 2018.

[2] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, 2006.

[3] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[4] M. Ghassemi, A. Sarwate, and R. Wright. Differentially private online active learning with applications to anomaly detection. In *Proc. Workshop on Artificial Intelligence and Security*, pages 117–128, 2016.

[5] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.

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

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

[8] M. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proc. Intl. Conf. Machine learning*, pages 65–72, 2006.

[9] S. Dasgupta. Two faces of active learning. *Theoretical computer science*, 412(19):1767–1781, 2011.

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

[11] S. Hanneke. Theoretical foundations of active learning. Technical report, Carnegie Mellon University, Pittsburgh, PA., 2009.

[12] M. Balcan. Active learning lecture notes. Technical report, Carnegie Mellon University, 2015.

[13] C.-C. Chang and C. J. Lin. Libsvm: A library for support vector machines. *ACM Trans. Intelligent Sys. and Tech.*, 2(3):27, 2011.

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

[15] B. Rubinstein, P. Bartlett, L. Huang, and N. Taft. Learning in a large function space: Privacy-preserving mechanisms for svm learning. *Journal of Privacy and Confidentiality*, 4(1):65–100, 2012.