Journal of Privacy and Confidentiality
Vol. 10 (2) 2020 TPDP 2018 Submitted March 2019 Published June 2020

UNDERSTANDING PRIVACY-UTILITY TRADEOFFS IN DIFFERENTIALLY PRIVATE ONLINE ACTIVE LEARNING

DANIEL M. BITTNER*, ALEJANDRO E. BRITO†, MOHSEN GHASSEMI*, SHANTANU RANE†, ANAND D. SARWATE*, AND REBECCA N. WRIGHT‡

⋆ Rutgers University
e-mail address: {daniel.bittner, mohsen.ghassemi, anand.sarwate}@rutgers.edu

† Palo Alto Research Center
e-mail address: {abrito, srane}@parc.com

‡ Barnard College
e-mail address: rwright@barnard.edu

Abstract. We consider privacy-preserving learning in the context of online learning. In settings where data instances arrive sequentially in streaming fashion, incremental training algorithms such as stochastic gradient descent (SGD) can be used to learn and update prediction models. When labels are costly to acquire, active learning methods can be used to select samples to be labeled from a stream of unlabeled data. These labeled data samples are then used to update the machine learning models. Privacy-preserving online learning can be used to update predictors on data streams containing sensitive information. The differential privacy framework quantifies the privacy risk in such settings. This work proposes a differentially private online active learning algorithm using stochastic gradient descent (SGD) to retrain the classifiers. We propose two methods for selecting informative samples. We incorporated this into a general-purpose web application that allows a non-expert user to evaluate the privacy-aware classifier and visualize key privacy-utility tradeoffs. Our application supports linear support vector machines and logistic regression and enables an analyst to configure and visualize the effect of using differentially private online active learning versus a non-private counterpart. The application is useful for comparing the privacy/utility tradeoff of different algorithms, which can be useful to decision makers in choosing which algorithms and parameters to use. Additionally, we use the application to evaluate our SGD-based solution and to show that it generates predictions with a superior privacy-utility tradeoff than earlier methods.

Key words and phrases: Differential Privacy, Active Learning, Anomaly Detection.

* Earlier versions of this work were presented at the 2018 ACM CCS Workshop on Theory and Practice of Differential Privacy [8] and at the 2016 ACM Workshop on Artificial Intelligence and Security [24].

JPrivacy Confidentiality
www.journalprivacyconfidentiality.org
DOI:10.29012/jpc.720

1. Introduction

The literature on differentially private data classification has grown rapidly to cover a variety of applications, algorithms and perturbation mechanisms [1, 19, 30, 43]. In this work, we are interested in a comparatively less investigated area: the incorporation of differential privacy into online learning schemes that frequently update the classifier models. In particular, we are interested in private online active learning wherein online classifier adaptation is achieved by diverting a fraction of the streamed samples to a human expert for labeling, and the classifier is updated using the human-labeled samples in a privacy-preserving manner.

Online learning differs from traditional machine learning in that data becomes available for training in a sequential manner rather than having initial access to the entire dataset. One rationale for using these online schemes is that a classifier that is dynamically updated to reflect the recent statistics of the streaming data can make more accurate predictions. An example of such a scenario is automatic screening of baggage at airports where the contents of passengers' baggage can vary due to weather conditions. A classifier that is updated according to newly streamed data is likely to be more accurate than a static classifier.

An important application area for the online learning setting is anomaly detection where data-driven machine learning approaches can yield effective anomaly detectors in a wide range of applications [10, 13, 29, 38, 56]. Classical hypothesis testing approaches for detecting anomalous data samples often rely on known or partially known statistical models of anomalous vs. non-anomalous instances. However, in many real-world applications, these models may not be known and so must be learned from data before deployment or may be refined in an online manner after deployment. For example, spam filters for email may be trained on an existing corpus and then tuned based on user feedback [39].

However, when these data streams in online applications contain personally identifiable information (PII) or sensitive personal information (SPI), privacy can be a significant concern. For example, auditing of financial transactions, fraud investigation in medical billing, and various national security applications entail sifting through sensitive information about individuals in order to find anomalous behaviors or entities. Therefore, it is imperative that the predictions or model parameters of the machine learning algorithm do not reveal information about the stream dataset used for learning the model. This problem is especially acute in the online setting where each update to the classification rule can potentially expose information about the data points that are used to perform the update. Differential privacy has been proposed by Dwork et al. [16] to address the fundamental question of the privacy risk incurred by publishing functions of private data. Since its introduction, differentially private algorithms have been proposed for a wide range of applications by several different research communities in the machine learning domain [1, 7, 11, 20, 28, 32, 41, 47, 49, 57]

In this work, we explore differentially private online active learning for scenarios where a "human in the loop" must do the labeling of selected samples. Active learning algorithms choose a subset of the data points using some selection criteria and only those data points are used to update the classifier. In anomaly detection problems, for example, where anomalies are relatively rare and the labeling of anomalies requires the human expert, the cost of acquiring a large labeled training set may be high. Thus, reducing the total number of labeled points needed can be particularly useful when the labeling costs may be prohibitively expensive. Additionally, by using only more informative points according to some informativeness metric, classifiers can potentially achieve improved accuracy. We

apply standard differential privacy techniques to control the privacy risk incurred by each of these steps at the cost of some loss in accuracy, or utility.

Many machine learning practitioners may lack a sense for the tradeoffs between privacy and accuracy entailed by the use of privacy preserving algorithms. It is challenging to understand how theoretical guarantees translate into practical application on different datasets, especially with a wide scope of parameter and model configurations. This lack of familiarity can make practitioners hesitant to adopt differentially private methods. One of the motivations of this work is to enable a human expert to understand how the different configurations of differentially private active learning algorithms can affect performance.

The contributions of this work are twofold:

- We propose an algorithm to perform a differentially private classification in the active learning setting.
- We provide an implementation and evaluation of a software system that can create, update, and release differentially private machine learning models using an online active learning modality

Our first contribution is a differentially private active learning algorithm. Information about the training stream is revealed when updating the classifier, selecting the points to be labeled, and choosing the timing of the updates. In other words, when using active learning in the online setting, both the selection rule and the update rule may reveal information about the training data. To privately update the classifier, we use the popular private stochastic gradient framework (SGD) [7, 14, 20, 49, 50] to perform stream-based online learning. For private selection under active learning, we modify a selection rule based on informative sample selection proposed by Tong and Koller [52]. Although privacy was not a consideration in the Tong-Koller paper, our modification enables us to leverage data point informativeness along with randomization to meet differential privacy guarantees.

Our second contribution is a software tool that enables exploratory analysis of our online learning system. Our tool allows analysts to develop their intuition about the tradeoffs between privacy, model adaptation and classification performance, and allow users to choose values for various parameters. Given the active learning component, it is necessary to enable the user to visualize the change in the classifier's performance as new data streams in, and to relate the classifier's behavior to the privacy parameters. In this work, we describe key aspects of the software architecture and the tool that we implemented to test and evaluate differentially private active learning schemes.

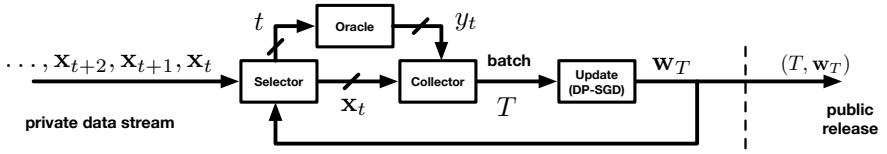


FIGURE 1. Block diagram of stream-based active screening and private updates. The dashed line indicates what is published by the algorithm.

1.1. **Algorithmic approach.** The general strategy of our algorithm is shown in Figure 1. We do not assume the samples are adaptive or can depend on the current value of the classifier. The goal is to learn a classifier which minimizes the expected risk over an (unknown)

distribution from which the data is drawn. The input data consists of a stream of unlabeled feature vectors whose labels (anomalous or non-anomalous) are known to an oracle (an expert who can label the points). At each time, the algorithm chooses whether to request a label from the oracle. Labeled points are collected over time and processed in batches, possibly at a slower rate. When a batch of labeled points is processed, the classification rule is updated and the time and the current classification rules are published. To select points for screening, the algorithm can use randomized response [54] or the exponential mechanism [37] to decide whether to select a given training sample. We instead use a modified version of the active learning heuristic of Tong and Koller [52] within the exponential mechanism [37] to improve the usefulness of the selected points for the learning task.

With randomized selection, the classifier update step could immediately use differentially private stochastic gradient descent (SGD) [7, 14, 49, 50]. However, due to the noise needed for differential privacy, the algorithm might potentially converge much more slowly in spite of the improved point selection process. Instead, to obtain better performance while still achieving privacy, we propose two mini-batching strategies, which we call fixed-length selection windows (FLSW) and fixed-length mini-batch (FLMB). The FLSW strategy computes model updates after fixed time intervals using a variable-sized mini-batch of points selected within the previous window. The FLMB strategy computes model updates after a pre-specified number of points has been selected. While FLSW uses variable memory and batch size, FLMB uses fixed memory and batch size. We evaluate both strategies and implement the one resulting in better predictor performance. The algorithm publishes updates to its detection rule as well as the time of the updates.

In Section 3 we provide a privacy analysis of our proposed algorithm. The privacy proofs follow from the properties of the basic differentially private mechanisms that we employ in our selection and update procedures. A utility analysis is more challenging in this setting: there is no analysis of the Tong-Koller method [52] even for the non-private setting. While some more recent works in the literature [5, 6] provide guarantees on the number of required queries or alternatively provide regret bounds [34], the heuristics used in this paper for point selection do not allow for straightforward application of such analysis techniques. We therefore leave providing theoretical utility guarantees as future work.

We give an empirical evaluation of tradeoffs between label complexity, privacy, and error for our method in the context of anomaly detection.

1.2. **Software System.** The software prototype system provides tools which enable decision makers to perform exploratory analysis of a private online learning system and develop intuitions about how differentially private classifiers perform in comparison to traditional counterparts. In Section 5 we describe the prototype in detail. The system provides detailed performance metrics for the classifier and its training. In particular, it updates these measures after each batch in the online processing environment to illustrate how selecting new data points affects the performance.

The system also allows users to test under different selectable conditions such as dataset, features, classifier and noise model, and training and privacy parameters. The tool currently implements linear support vector machine (SVM) and logistic regression algorithms in both their private and non-private versions. Other kinds of classification algorithms can be readily added as long as differentially private mechanisms can be designed for their static (i.e., non-active) versions. Users can tune specific model parameters for privacy, optimization, batch size, and sample selection configurations.

This system will enable machine learning practitioners to better understand the practical implications of differential privacy. It allows practitioners to develop intuitions about the tradeoffs between accuracy and privacy and how different configurations can affect the ratios of these tradeoffs. The goal of the system is to encourage practitioners to feel more confident about the results of adopting differentially private systems.

2. Preliminaries

We consider a model, shown in Figure 1, in which a collection $\mathcal{D}_x = \{\mathbf{x}_t \in \mathbb{R}^d : t = 1, 2, \ldots\}$ of samples is presented to a learning algorithm in an online manner. We assume all data points are bounded, so

$$\|\mathbf{x}_t\| \leq M. \tag{2.1}$$

Associated with each sample point $\mathbf{x}_t$ is a (hidden) label y_t, which is known to an oracle $\mathcal{O}$. In the anomaly detection context, the vector $\mathbf{x}_t$ specifies the value of measured features about an event or individual observed at time t, and the label $y_t \in \{-1, +1\}$ indicates whether the sample is anomalous. Let $\mathcal{D} = \{(\mathbf{x}_t, y_t)\}$ be the labeled dataset. The feature vectors, but not the labels, are given to the online learning algorithm. The algorithm must decide at each time t whether to query the oracle for the label of $\mathbf{x}_t$ or discard it before observing the next sample. The goal of the algorithm is to learn a classifier $\mathbf{w}$ that assigns a correct label $\hat{y}$ to a feature vector $\mathbf{x}$. The labels are assigned by applying the sign function to the inner product of $\mathbf{w}$ and $\mathbf{x}$, i.e. $\hat{y} = \operatorname{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle)$.

The state of the algorithm is given by the current classifier $\mathbf{w}$. At each time $t = 1, 2, \ldots, N$, the state is $\mathbf{w}_t$ and the algorithm makes two choices. First, it selects whether or not to request the label y_t from the oracle. Second, it can choose to update the classifier/state $\mathbf{w}_t$ and publish the resulting update. We consider models in which $\mathbf{w}_t$ is updated based on new labeled data; updates are either based on a fixed schedule (if there is no new labeled data the algorithm publishes the previous state) or based on the number of new labeled points. We call the time between updates a *selection window*. The algorithm requires a buffer that keeps the labeled points during each selection window until the end of the window.

The overall output of the algorithm over n iterations is the sequence of classifier estimates $\{\mathbf{w}_t : t = 1, 2, \ldots, n\}$ or alternatively the pairs of update time and update $\{(T_i, \mathbf{w}_{T_i}) : i = 1, 2, \ldots\}$, which determine the times $\{T_i\}$ when the algorithm chooses to update the classifier.

Support vector machine: We use support vector machine (SVM) to find a classifier by approximately solving the following regularized risk minimization problem:

$$\min_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w}) = R(\mathbf{w}) + \mathbb{E}_{\mathcal{P}} \left[\ell_h(\mathbf{w}; (\mathbf{x}, y)) \right], \tag{2.2}$$

where $(\mathbf{x}, y)$ is a sample-label pair such that $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathcal{Y} = \{-1, 1\}$, $\mathcal{P}$ is the underlying joint distribution of the pair $(\mathbf{x}, y)$, and $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2$ is a regularizer. The loss function ℓ_h is defined as the hinge loss $\ell_h(\mathbf{w}; (\mathbf{x}, y)) = [1 - y \langle \mathbf{w}, \mathbf{x} \rangle]_+$, which is convex. Throughout this paper, we use words sample, data point, and instance interchangeably to refer to $\mathbf{x}$.

For a training set of labeled points $\mathcal{S}$ (which in our case will be the subset of the data points whose points have been labeled), the SVM algorithm minimizes the regularized

empirical risk as a proxy for (2.2):

$$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}) = R(\mathbf{w}) + \frac{1}{|\mathcal{S}|}\sum_{(\mathbf{x},y)\in\mathcal{S}} \ell_h(\mathbf{w};(\mathbf{x},y)). \tag{2.3}$$

In the online setting, we assume that the true distribution $\mathcal{P}$ is unknown to the machine learning algorithm. Instead, the learner is presented sequentially with samples $\{\mathbf{x}_i : i = 1, 2, \ldots\}$ drawn from the marginal distribution $\mathcal{P}_X$. The regularized empirical risk minimization framework includes many classification and regression problems, we use SVM for simplicity of exposition.

Differential privacy: Differential privacy [16] is a statistical method for measuring the privacy risk from publishing functions of private data. In our setting, the private data is the set of pairs $\{(\mathbf{x}_t, y_t) : t = 1, 2, \ldots\}$. For a finite time horizon n, consider two data streams that differ in a single point, say at time κ:

$$\mathcal{D} = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_\kappa, y_\kappa), (\mathbf{x}_{\kappa+1}, y_{\kappa+1}), \ldots, (\mathbf{x}_n, y_n)$$
$$\mathcal{D} = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}'_\kappa, y'_\kappa), (\mathbf{x}_{\kappa+1}, y_{\kappa+1}), \ldots, (\mathbf{x}_n, y_n),$$

where all data vectors satisfy (2.1). We call such databases neighboring. We call a randomized algorithm $\mathcal{A}$, operating on $\mathcal{D}$, ϵ-differentially private if the presence of any individual data point does not change the probability of the algorithm output $\mathcal{A}(\mathcal{D}) \in \mathcal{C}$ by much, for any set of $\mathcal{C}$. Formally, we say algorithm $\mathcal{A}$ provides ϵ-differential privacy if for any two neighboring databases $\mathcal{D}$ and $\mathcal{D}$' and any set of outputs $\mathcal{C}$,

$$\left|\log \frac{\mathbb{P}(\mathcal{A}(\mathcal{D}) \in \mathcal{C})}{\mathbb{P}(\mathcal{A}(\mathcal{D}') \in \mathcal{C})}\right| \leq \epsilon, \tag{2.4}$$

where $\mathcal{A}(\mathcal{D})$ is the output of $\mathcal{A}$ on the dataset $\mathcal{D}$.

For the online algorithms described above, the output set would consist of n-tuples of classifiers $\{\mathbf{w}_t : t = 1, 2, \ldots, n\}$. Thus, we are interested in controlling the total amount of privacy risk ϵ due to both the selection of points for updating the classifiers as well as the updated classifiers. In the differential privacy threat model, an adversary observes the output of the algorithm and attempts to infer whether the outcome came from input $\mathcal{D}$ or $\mathcal{D}'$; successful adversarial inference would mean that the adversary would learn whether $(\mathbf{x}_\kappa, y_\kappa)$ or $(\mathbf{x}'_\kappa, y'_\kappa)$ was in the data stream. The parameter ϵ controls how difficult this hypothesis test is, bounding the tradeoff between false alarm and missed detection (Type I and Type II) errors [31, 55].

Privacy-preserving stream-based learning: Because our procedure reveals the classifiers over time, an adversary observes both the updates and the timing of updates. This means that potentially, the selection of points whose labels we query as well as the classifier updates must guarantee a total ϵ-differential privacy. To gain insight, let us consider two extremes.

The first example in Algorithm 1 simply asks for all labels and sets corresponding classifiers to zero. Next, it performs one update at time n using a batch ϵ-differentially private SVM training method such as objective perturbation [11]; we call this DPERM in Algorithm 1. Since the algorithm selects all points, the point selection process trivially guarantees differential privacy, and the SVM training is differentially private from previous results [11]. This approach has high label complexity and high computation cost from training an SVM on the entire dataset at time n.

Algorithm 1 Batch SVM

Input: ϵ, stream $\{\mathbf{x}_t\}$.
Initialize: $\mathbf{w}_0 = \mathbf{0}$, $\mathbf{S}_0 = \emptyset$.
for $t = 1$ to n **do**
  Ask oracle $\mathcal{O}$ for label y_t of $\mathbf{x}_t$.
  $\mathcal{S}_t \leftarrow \mathcal{S}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$.
  Set $\mathbf{w}_t = \mathbf{w}_{t-1}$.
end for
Update $\mathbf{w}_t$ using $\mathsf{DPERM}(\mathcal{S}_n, \epsilon)$.
Output $\mathbf{w}_t$.

Algorithm 2 Private Stochastic Gradient Descent

Input: ϵ, stream $\{\mathbf{x}_t\}$, step sizes $\{\eta_t\}$
Initialize: $\mathbf{w}_0 = \mathbf{0}$
for $t = 1$ to n **do**
  Ask oracle $\mathcal{O}$ for label y_t of $\mathbf{x}_t$.
  Update $\mathbf{w}_t$ using $\mathsf{DPSGD}(\mathbf{w}_{t-1}, \mathbf{x}_t, y_t, \eta_t, \epsilon)$.
  Output $\mathbf{w}_t$.
end for

On the other end of the update-frequency spectrum, we have Algorithm 2, which simply performs noisy stochastic gradient updates that guarantee differential privacy [50]. Again, this algorithm asks for the label of every data point and the updates are performed in a differentially private way, so the overall algorithm guarantees ϵ-differential privacy. This approach has low per-iteration complexity.

In settings where the learning algorithm is processing private or sensitive data, we would like to limit the privacy risk by not querying the labels of many points. It is also beneficial to limit the number of labels that must be queried in cases where training labels for anomaly detection must be generated by costly experts. In the next section, we describe how to use ideas from active learning to design algorithms that trade off label complexity, privacy, and classifier accuracy in this setting.

3. SVM Private Active Learning

In our system model, both the sample selection and classifier update must be made differentially private. In anomaly detection problems we are concerned with label complexity. When selecting samples for labeling, we would like to select points that are informative. We use a heuristic introduced by Tong and Koller [52] for training a support vector machine (SVM) using active learning. Their approach uses an informativeness measure based on the closeness to a hyperplane estimate $\mathbf{w}$. In our model, the informativeness of point $\mathbf{x}$ with respect to a hyperplane $\mathbf{w}$ is measured by its closeness to the hyperplane. Let

$$d(\mathbf{x}, \mathbf{w}) \triangleq \frac{|\langle \mathbf{w}, \mathbf{x} \rangle|}{\|\mathbf{w}\|} \tag{3.1}$$

The informativeness is

$$c(\mathbf{x}, \mathbf{w}) = \exp\left(-d(\mathbf{x}, \mathbf{w})\right) \in [0,\ 1].$$

Tong and Koller originally suggested this method for active learning in the pool-based setting: at iteration t the learner would select the point $\mathbf{x}_i$ with the largest informativeness $c(\mathbf{x}_i, \mathbf{w})$ and ask the oracle for the label. The algorithm then estimates a new $\mathbf{w}$ by retraining an SVM on the entire data set.

In our model, samples come in sequentially, so we define

$$c(t) = c(\mathbf{x}_t, \mathbf{w}_t) \tag{3.2}$$

at time t. In a non-private selection strategy, if the informativeness of sample $\mathbf{x}_t$ meets a certain threshold τ, then the learner queries the oracle to obtain y_t and updates the classifier. Otherwise, it discards $\mathbf{x}_t$ and waits for the next instance while setting $\mathbf{w}_{t+1}{=}\mathbf{w}_t$. Note that the condition $c(t) > \tau$ is equivalent to $d(\mathbf{x}_t, \mathbf{w}_t) < \log \frac{1}{\tau}$, which means that the sample $\mathbf{x}_t$ is within a *selection slab* of width $2 \log \frac{1}{\tau}$ around $\mathbf{w}_t$. We call this an *online active learning* strategy.

3.1. **Differentially Private Point Selection.** As mentioned earlier, the active learning algorithms we propose here make two decisions at each time t: first, whether to request the label y_t and add $(\mathbf{x}_t, y_t)$ to the training set, and secondly, whether to update the classifier so that $\mathbf{w}_{t+1} \neq \mathbf{w}_t$. We refer to our selection strategy as an online T-K (Tong-Koller) strategy. The original T-K heuristic's decision to query was deterministic, so to guarantee privacy we must randomize both the decision to query and the gradient step. At time t, based on the current state $\mathbf{w}_t$, the algorithm selects $\mathbf{x}_t$ to be labeled based on whether it passes the informativeness threshold. We can therefore use randomized mechanisms to select whether to label the point or not. When the algorithm decides to update $\mathbf{w}_t$ we can use private stochastic gradient descent (SGD) [7, 50].

Bernoulli selection: The simplest approach is to compare the informativeness $c(t)$ to a threshold and then use randomized response [54] to select the point. More formally, for parameters $p > 1/2$ and τ, the selection variable $s_t \sim \mathsf{Bern}(p)$ if $c(t) \geq \tau$ and $s_t \sim \mathsf{Bern}(1-p)$ if $c(t) < \tau$. Standard arguments imply that this provides $\epsilon_{\mathsf{ber}} = \log \frac{p}{1-p}$ differential privacy (Lemma 4.1).

Exponential selection: A strategy that is more in the spirit of the Tong-Koller method uses the exponential mechanism [37]. Consider a threshold on $d(\cdot, \cdot)$, so that we consider $d(\cdot, \cdot) \leq b$ and $d(\cdot, \cdot) > b$ as separate cases. Within the selection slab defined by b, the algorithm selects points with constant probability. Outside the slab it selects with probability that decays exponentially with the distance.

Let

$$\begin{aligned} q(t) &= \begin{cases} e^{-b\epsilon/\Delta} & d(\mathbf{x}_t, \mathbf{w}_t) \leq b \\ e^{-d(\mathbf{x}_t, \mathbf{w}_t)\epsilon/\Delta} & d(\mathbf{x}_t, \mathbf{w}_t) > b \end{cases} \\ &= \exp\left(-\max\left\{b, d(\mathbf{x}_t, \mathbf{w}_t)\right\} \cdot \epsilon/\Delta\right) \end{aligned} \tag{3.3}$$

where $\Delta = (1 - \frac{b}{M})M$ and $\epsilon > 0$. For this strategy, $s_t \sim \mathsf{Bern}(q(t))$. In this way, every point has a chance of being selected, so the adversary cannot infer whether an observed point is inside the selection slab or not. However, more informative points are still more likely to be selected. The privacy guarantee of this method is given in Lemma 4.2.

3.2. **Differentially Private Update.** In addition to making the point selection private, we must also make the update step private. We consider two strategies based on mini-batching: the fixed-length selection window (FLSW) policy, which updates after a fixed number of time steps, and the fixed-length mini-batch (FLMB) policy, which updates after a fixed number of selected points. Both strategies use a mini-batch update rule. Mini-batching is a well-known method in stochastic optimization for machine learning to control the variance of the subgradient estimates. In a privacy preserving algorithm, it also provides ambiguity in the contribution of each member of the mini-batch to the approximate gradient [49].

To retain the privacy of the users whose data are present in $\mathcal{D}$, during gradient update step, one method is to follow the differentially private SGD update [50]. In this method, a controlled noise term $\mathbf{z}_t$ is added to each update. More specifically, in order to make updates ϵ_g-differentially private, we add a random noise vector $\mathbf{z}_t \in \mathbb{R}^d$ drawn i.i.d. from the following probability distribution:

$$\mathbb{P}(\mathbf{z}) = \gamma e^{-(\epsilon_g/2M)\|\mathbf{z}\|} \tag{3.4}$$

where M is an upper bound on every $\mathbf{x}_i \in \mathcal{D}_X$ and γ is a normalizing constant.

For a labeled point $(\mathbf{x}, y)$ let

$$u = \mathbb{1}(\ell_h(\mathbf{w}; (\mathbf{x}, y)) > 0) \tag{3.5}$$

be the indicator that the hinge loss is positive. For a batch $\mathcal{B} = \{(\mathbf{x}_t, y_t)\}$ of $B = |\mathcal{B}|$ points, the differentially-private mini-batch update rule is given by

$$\mathbf{w}' = \mathbf{w} - \eta \left(\lambda \mathbf{w} - \frac{1}{B} \sum_{t \in B} y_t \mathbf{x}_t u_t + \frac{1}{B} \mathbf{z} \right) \tag{3.6}$$

where $\mathbf{z} \sim \mathbb{P}(\mathbf{z})$ in (3.4), u_t is given by (3.5), η is the learning rate, and λ is the regularization parameter.

Fixed-length selection windows (FLSW):. In this update method, the learner collects labeled samples into a batch $\mathcal{B}$ during an interval of length N where the data stream has length $n > N$. It updates the classifier at a rate slower than it observes incoming samples: once every N observations. Since the adversary can only see the updates in $\mathbf{w}_t$ when $t \mod N = 0$, we can take advantage of the ambiguity in the number of the samples selected during the N-length interval. An extreme version of the FLSW rule is the *immediate update rule*, which takes $N = 1$.

Fixed-length mini-batches (FLMB):. Here, before updating the classifier, the algorithm waits until a mini-batch of L labeled instances are collected. As before, the algorithm needs to decide whether or not to select a given sample for labeling immediately after it observes the sample. Thus the batch $\mathcal{B}$ has size $B = L$ in update rule (3.6). Unlike the FLSW method, the number of labeled samples in a batch is fixed. However, the adversary can observe the number of samples that are observed by the algorithm in each interval before L samples are selected. For the first $L - 1$ samples in the batch, the selection result (to label or not to label) for each observation is not available to the adversary, and this ambiguity in the index of the labeled samples provides privacy. By construction, the index – but not the value – of the L^{th} labeled sample is indeed visible to the adversary. Privacy guarantees for FLMB are discussed in the next section.

Algorithm 3 FLSW update with randomized screening

Input: ϵ_s, ϵ_g, stream $\{\mathbf{x}_t\}$, step sizes $\{\eta_t\}$, λ, N, batch $\mathcal{B}$.
Initialize: $\mathbf{w}_0 = \mathbf{0}$ and $\mathcal{B} = \emptyset$.
for $t = 1$ to n **do**
  Use ϵ_s-DP method and $\mathbf{w}_{t-1}$ to decide whether to ask oracle $\mathcal{O}$ for label y_t of $\mathbf{x}_t$.
  if y_t labeled by $\mathcal{O}$ **then**
    Add $(\mathbf{x}_t, y_t)$ to $\mathcal{B}$.
  end if
  if $t \mod N = 0$ **then**
    Update $\mathbf{w}_t$ using (3.6) with $\mathbf{w}_{t-1}$, ϵ_g, η_t, λ and batch $\mathcal{B}$.
    Set $\mathcal{B} = \emptyset$.
    Output $\mathbf{w}_t$.
  else
    Set $\mathbf{w}_t = \mathbf{w}_{t-1}$.
  end if
end for

Algorithm 4 FLMB update with randomized screening

Input: ϵ_s, ϵ_g, stream $\{\mathbf{x}_t\}$, step sizes $\{\eta_t\}$, λ, L, batch $\mathcal{B}$.
Initialize: $\mathbf{w}_0 = \mathbf{0}$ and $\mathcal{B} = \emptyset$.
for $t = 1$ to n **do**
  Use ϵ_s-DP method and $\mathbf{w}_{t-1}$ to decide whether to ask oracle $\mathcal{O}$ for label y_t of $\mathbf{x}_t$.
  if y_t labeled by $\mathcal{O}$ **then**
    Add $(\mathbf{x}_t, y_t)$ to $\mathcal{B}$.
  end if
  if $|\mathcal{B}| = L$ **then**
    Update $\mathbf{w}_t$ using (3.6) with $\mathbf{w}_{t-1}$, ϵ_g, η_t, λ and batch $\mathcal{B}$.
    Set $\mathcal{B} = \emptyset$.
    Output $\mathbf{w}_t$.
  else
    Set $\mathbf{w}_t = \mathbf{w}_{t-1}$.
  end if
end for

4. Analysis

We now quantify the differential privacy guarantees under our models and update procedures. We compare the utility performance of the different methods empirically. As mentioned in the introduction, providing theoretical guarantees in terms of label complexity for our method is challenging. Some works in the literature (see e.g. [5, 6]) obtain such guarantees by carefully quantifying the reduction in the version space at every step of the point selection procedure of their proposed algorithms. This approach is however not applicable to our method since our subset selection rules do not permit a clear analysis of the version space reduction. The main idea behind the selection rules in this paper is the following: since the current SVM hyperplane $\mathbf{w}_t$ is the center of the largest hypersphere that can fit inside the current version space [52], points closer to current classifier are more "likely" to halve the

current version space (see Tong and Koller [52] for a more detailed discussion). However, even in the non-private scenario, we cannot quantify the reduction in the version space. This is even more challenging in the differentially private scenarios where we introduce randomness to the point selection rule. Therefore, we leave theoretical analysis of the label complexity of our methods as an open question.

4.1. **Privacy Analysis for Individual Steps.** In the proofs of the theorems in this section, let $s_t \in \{0, 1\}$ indicate whether the algorithm queries the label of the t^{th} point. We begin with privacy guarantees for the selection steps and mini-batch steps that make up the algorithm.

Bernoulli selection strategy: Lemma 4.1 bounds the privacy risk of the Bernoulli selection strategy.

Lemma 4.1. *Consider an online active learning algorithm with the Bernoulli selection strategy as described in Section 3.1 with $p > \frac{1}{2}$. Assuming that the most recent value of $\mathbf{w}_t$ is public, this query strategy is ϵ_{ber}-differentially private, where $\epsilon_{\mathsf{ber}} = \log\left(\frac{p}{1-p}\right)$.*

Please refer to Appendix A.1 for the proof.

Exponential selection strategy: Lemma 4.2 presents the privacy guarantee provided by the exponential mechanism.

Lemma 4.2. *Consider an online active learning algorithm with the exponential selection strategy as described in Section 3.1. Suppose that b in (3.3) satisfies $\exp(-b\epsilon_{\mathsf{exp}}/\Delta) \leq 1/2$. Assuming that the most recent value of $\mathbf{w}_t$ is public, this query strategy is ϵ_{exp}-differentially private.*

Please refer to Appendix A.2 for the proof.

Mini-batch update:

Lemma 4.3. *The gradient step in (3.6) with batch $\mathcal{B}$ is ϵ_g-differentially private.*

Please refer to Appendix A.3 for the proof.

4.2. **Immediate Update.** As a baseline, we analyze the FLSW scheme with window size $N = 1$, which corresponds to immediate updates. Standard composition theorems [31, 36] allows us to find the overall differential privacy guarantee under each scenario under either Bernoulli or exponential sampling.

Corollary 4.4. *Each update in Algorithm 3 with $N = 1$ is ϵ_g-differentially private.*

Proof. The proof follows from Lemma 4.3 with batch size $B = 1$. □

Theorem 4.5. *The FLSW algorithm in Algorithm 3 with $N = 1$ is $(\epsilon_s + \epsilon_g)$-differentially private.*

Please refer to Appendix A.4 for the proof. The immediate update strategy provides a baseline performance against which to compare the two mini-batch strategies in Algorithms 3 and 4.

4.3. **Mini-batch Update with FLSW.** Under the FLSW policy, updates happen every N observations. Privacy stems from the fact that the individual point selections within this interval of length N are hidden, and only the updates $\{\mathbf{w}_{kN}\}$ are revealed.

Selection step: Each update of the classifier in the FLSW algorithm consists of N differentially private selection steps. The only conclusion an adversary can make with regards to the selection process by observing the decision vectors $\{\mathbf{w}_{kN}\}$, is whether any samples have been selected in the k^{th} window or not. Therefore, we consider only two events: one where no sample has been selected at all, i.e., $S_k = 0$, and the other where at least one sample is selected to be queried for its label, i.e., $S_k = 1$. First, consider the case where $\mathbf{w}_{(k+1)N} = \mathbf{w}_{kN}$, which means that $s_t = 0$ for all $kN \leq t \leq (k+1)N$. In this worst case scenario, there is no ambiguity as to which points have been selected during the given window. Thus, the privacy guarantee for $S_k = 0$ is the same as when we use the immediate update.

When $S_k = 1$, the information revealed is always less than when the selection result for every observation is available to the adversary, which happens in the immediate update scenario. Consequently, the privacy guarantee here is also no worse than that of the selection strategies in the immediate update method, which means that the entire selection step in the FLSW mini-batch method is ϵ_s-differentially private.

Gradient step: The following Corollary states the privacy guarantee provided for a mini-batch gradient update.

Corollary 4.6. *The gradient update step in Algorithm 3 (FLSW) is ϵ_g-differentially private.*

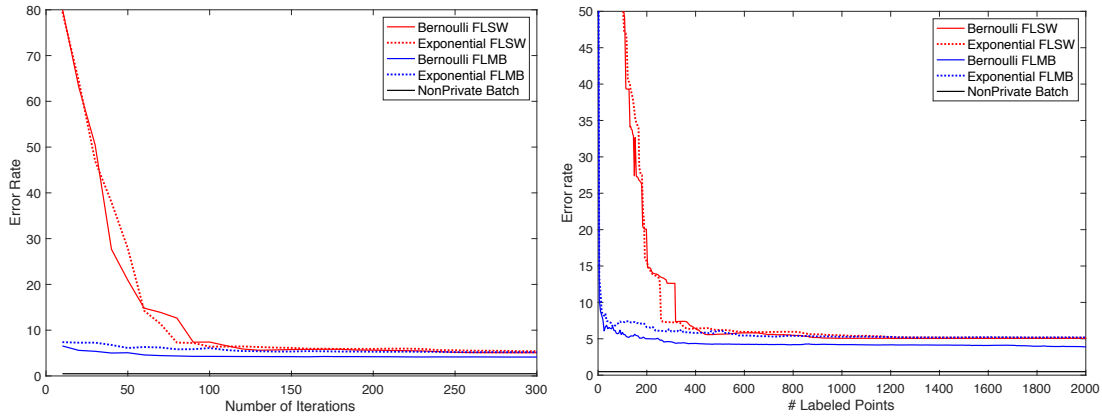
Proof. The proof follows from Lemma 4.3 with batch size B_k. □

Note that the guarantee in this Corollary is ϵ_g-DP by design of the gradient step. Because each step averages B_k points, the effective noise per sample is reduced by a factor of B_k. Since the gradient step is less noisy, the performance of the algorithm should improve, empirically. However, because the batch size varies in each length-N epoch, the noise level will also be variable. As we will see, this variability can affect the empirical performance of the FLSW method. For larger N the expected batch size $\mathbb{E}[B_k]$ will be larger, so longer windows can help improve performance at the expense of a (possibly) larger buffer size for the learning algorithm.

4.3.1. *Overall Privacy Guarantees.*

Theorem 4.7. *The FLSW algorithm in Algorithm 3 with general N guarantees at most $(\epsilon_s + \epsilon_g)$-differential privacy.*

Please refer to Appendix A.5 for the proof.



(A) Error vs. #iterations (B) Error vs. #labels requested

FIGURE 2. Classifier performance as a function of number of iterations and the number of labels requested for the `KDD Cup 1999` dataset.

4.4. **Mini-batch Update with FLMB.** As mentioned earlier, the adversary can see the number of data points observed by the learner before it collects L labeled points. By randomizing the selection process and the gradient update rule, we guarantee differential privacy of the FLMB mini-batch algorithm. In this section, we discuss this guarantee.

Selection step: In FLMB, the adversary does not discover whether any sample is selected or not, but finds out *how many* samples are observed before L samples are selected for re-labeling and updating the model. For the L^{th} sample in the batch, the adversary discovers the *index* of the data stream at which the batch $\mathcal{B}$ was completed. Privacy results from the differential privacy model, in which the adversary does not know whether the *value* of any given sample in the retraining batch $\mathcal{D}$ is the same as, or different from, the value of the corresponding sample in his adjacent data stream $\mathcal{D}'$. This argument applies also to two corner cases, namely (1) $L = 1$ (2) L sample points are observed and all of them are selected. For all other cases, parameters of the Bernoulli or exponential selection strategies can be chosen to provide ϵ_s-DP during the sample selection process, based on Lemmas 4.1 and 4.2.

Gradient step: The FLMB method guarantees a fixed batch size of L samples per iteration. Thus for the same privacy guarantee ϵ_g, it uses a factor L less noise.

Corollary 4.8. *The gradient update step in in Algorithm 4 is ϵ_g-differentially private.*

Proof. The proof follows from Lemma 4.3 with batch size L. □

4.4.1. *Overall Privacy Guarantees.*

Theorem 4.9. *The FLMB algorithm in Algorithm 4 with batch size L guarantees at most $(\epsilon_s + \epsilon_g)$-differential privacy.*

Proof. Please refer to Appendix A.6 for the proof.

4.5. **Evaluation of Our Theoretical Approach.**

Dataset and preprocessing: We apply our classification methods on the `KDD Cup 1999` dataset [40], which contains information about network intrusion detection and closely matches our anomaly detection motivation. For this paper, our dataset is composed of about 494,000 samples, out of which the first 400,000 samples are used to extract training and validation data, while the final 94,000 samples are used as the test dataset(also called the evaluation dataset). The task is to build an intrusion detection model that can differentiate "normal" network connections from "intrusions" or "attacks". As a preprocessing step, we remove two irrelevant features (`numoutbound_cmds` and `is_host_login`) and convert 7 remaining categorical features into binary vectors. This process leaves us with 120 features. We then project all entries of both data sets onto the unit ball.

Evaluation procedure: In this section, we set differential privacy guarantees $\epsilon_g = 1$ and $\epsilon_s = 1$. We set $p = e/(1+e)$ to make the Bernoulli selection processes also 1-differentially private. Parameter b in (3.3) is set to 0.2. We used step sizes $\eta_t = \eta/t$ and found values of η and the regularization parameter λ using cross-validation[1]. Unless stated otherwise, the threshold τ in Bernoulli strategies is set to $e^{-0.2} \approx 0.8$. At every classifier update iteration we add noise according to (3.4), and the iterate is projected onto the uniform ball. We averaged the results obtained over 10 random permutations of training data streams. The error bars in Figure 3 are 1-standard deviation error bars.

Error rate over time: Fig. 2a shows the misclassification error rates of FLSW and FLMB update methods. For both FLSW and FLMB, we set the selection window size to 5. We observe that even though all of the methods show fast convergence to their limit value, due to the added gradient noise we see a small gap between the non-private batch error rate and the limit error rates of our proposed methods. After processing many samples, progress slows due to the small step size in the SGD iterations. This shows the trade-off between privacy and accuracy in our algorithms. We observe that, in general, the FLMB performance tends to be better than the FLSW performance. We conjecture that the fixed batch size controls the variance of the subgradient steps in the SGD iterations.

Error rate versus label cost: A comparison between different methods based on Fig. 2a would not be fair as each method uses a different number of labeled points. Therefore we compare the error rates against number of labeled points. Fig. 2b shows the misclassification error rates as a function of label costs, assuming that each label costs 1 unit to acquire. Observe that the FLMB method is more efficient than FLSW in the sense that it achieves better error rates for a given label budget. We believe this is because for our choice of parameters, the number of labeled points per iteration, denoted by B in (3.6), in FLSW is smaller than or equal that of FLMB. This means, according to update rule (3.6), larger mini-batches are selected and smaller noise is added to the iterate during an FLMB update.

We note that in our FLSW experiment in this section, the values on the label complexity axis (x-axis) are not equally spaced and the spaces also differ across experiments over different permutations. In order to address this issue, we use piece-wise linear interpolation and sample the interpolated iterate values at the desired equally spaced points.

[1]We did not use differential privacy to select these parameters, although this approach could also be used as suggested by Chaudhuri and Vinterbo [12]

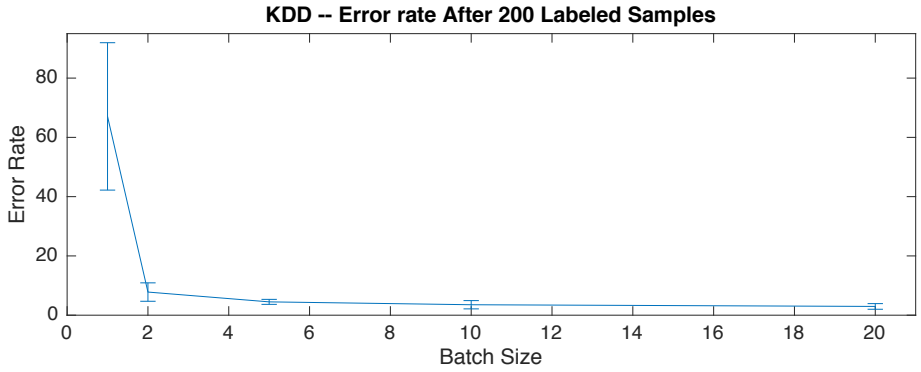
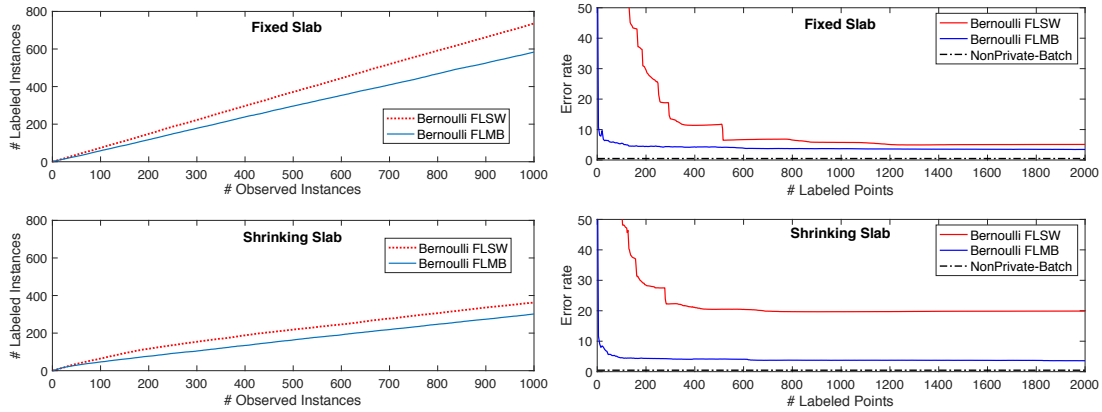


FIGURE 3. Classifier performance as a function of batch size for FLMB with Bernoulli strategy



(A) #labels vs #iterations for fixed & time-varying Bernoulli strategy in FLMB for KDD Cup 1999

(B) Error versus #labels for fixed & time-varying Bernoulli strategy in FLBM for KDD Cup 1999

FIGURE 4. Comparing fixed and shrinking slabs for different strategies in terms of labels complexity and error performance.

Selection method: Surprisingly, we see from the experiments in Figures 2a and 2b that the difference between the Bernoulli and Exponential selection methods is negligible in terms of overall performance. We conjecture two reasons for this. First, by choosing a large value of ϵ_s for the selection step, the difference between the two sampling distributions is not too significant. Second, the active learning heuristic provides the most improvements in the early stages of the classifier training process: once the classifier has stabilized, additional training samples do not improve the performance significantly. Noting the slight superiority of the FLMB algorithm with Bernoulli sampling, we will use that combination in more detailed experimentation described in Section 5.

Effect of mini-batching: Fig. 3 shows the error rate of the FLMB update method with Bernoulli selection strategy for batch sizes $\{1, 2, 5, 10, 20\}$. The results are from 200 labeled examples. We observe that mini-batching, as expected, reduces the variance in results.

Time-varying selection strategies: Label costs are not necessarily constant over time. It can be the case that the more the learner queries for labels, the more the oracle charges per label. Moreover, as the classifier is updated using more labeled points, it becomes more accurate and points far from the classifier are more likely to be noisy. Therefore, we would like to lower the chances of labeling uninformative, noisy data points by making the selection policy stricter.

As an example, we study the performance of both FLSW and FLMB methods with time-varying Bernoulli selection strategy over the `KDD Cup 99` dataset. We compare using a fixed threshold $\tau = e^{-0.2} \approx 0.8$ versus a time-varying threshold $\tau = e^{-\frac{1}{m}}$. The latter case uses a selection slab that shrinks linearly over time. Here, m is the counter for classifier updates, not the data stream counter. The results of this experiment are given in Figures 4a and 4b. Fig. 4a shows a considerable reduction in the label complexity in the time-varying query method. In fact, we observe that the selection rate is not linear anymore and becomes sublinear.

As shown in Fig. 4b, the generalization performance of the FLMB method, for given number of labeled points, is not compromised. This is not surprising since each update uses a fixed mini-batch size and the learner is simply more selective about informativeness and needs to wait for a longer time to collect a batch. The results show that this selection strategy is actually querying more informative points and does not lose much by discarding more points. We expect that for some datasets this strategy actually could improve the performance by using its labeling budget more judiciously.

On the other hand, the classification performance of the FLSW algorithm is considerably compromised. We suspect that with a shrinking selection slab the bar for selection is too high; so few instances can pass the informativeness threshold that this method (in later iterations) effectively loses the "mini-batch" property.

5. Software Prototype for Privacy-Aware Online Active Learning

We built a software tool to facilitate analysis of the proposed differentially private anomaly detection approach. In particular, because this approach has various design variables, our goal is to enable researchers and engineers to evaluate the effect of changing privacy parameters (e.g., ϵ_s, ϵ_g, batch size, informativeness metrics, visualization parameters) as well as the classifier parameters (e.g., learning rate and regularization parameters) on the performance of anomaly detection and on the evolution of the learned classifiers.

5.1. Proposed Architecture.

The software architecture of the proposed differentially private active learning system is depicted in Fig. 5. The initial training data set as well as the streaming data to be classified is assumed to reside in separate databases, which could be mySQL database or csv files. The learning component of the system consists of an Application Programming Interface (API) that enables the use of available fast numerical computation and machine learning libraries, such as scikit-learn, Theano and TensorFlow. In our tool prototype, we use csv files and the Theano library. To perform the training in a differentially private manner, we selected the algorithms for sample selection and classifier update that showed better performance as reported in Section 4.5. While we describe a version of the prototype based on the differential privacy methods proposed in Section 3, the architecture accommodates other well-known differential privacy mechanisms. These include adding noise directly to the data (input perturbation), or to the trained classifier

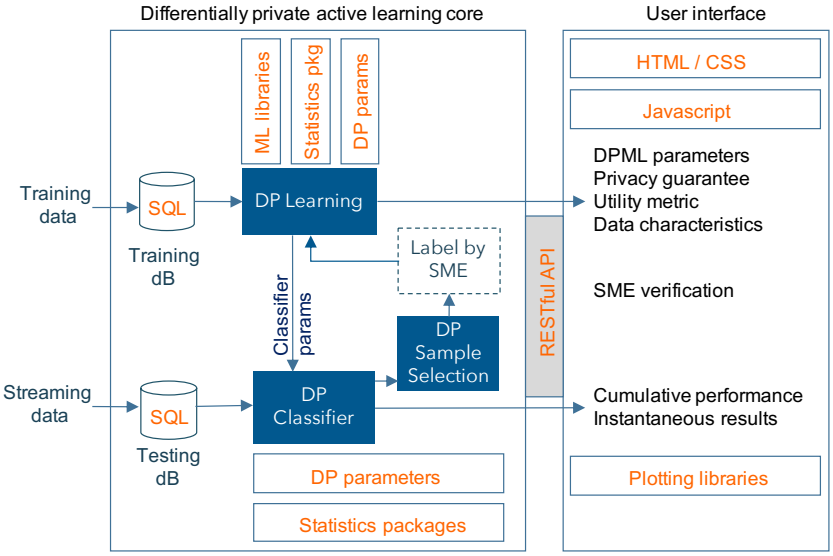


FIGURE 5. Software architecture of the proposed differentially private active learning system.

parameters (output perturbation), or to intermediate quantities such as objective functions (objective perturbation) or gradient calculations [48]. The architecture supports privacy parameters of noise distributions, which are chosen based on the desired level of privacy. Privacy parameters such as ϵ and δ are presented as inputs to the learning component.

The architecture defines a visual interface (shown in the diagram of Fig. 5) in order to provide a means for choosing the dataset and attributes to be processed, to specify the settings for the classification algorithm, the differential privacy parameters, the schedule of model updates, and other input settings as well as to report status and progress of the learning tasks. An example user interface implementing the architecture visual interface is presented in the next section (See Fig. 6). Finally, the architecture supports streaming of input data for classifiers which predict corresponding class labels. These predictions are used for the presentation of classification results in various forms, for example: (1) a plot of the instantaneous class label against time for the streaming data; (2) a constantly updating readout of the accuracy of the classifier calculated over batches of the streaming data; (3) a histogram of class labels; and (4) various metrics including accuracy, precision, recall, etc.

5.2. **Active Learning Tool.** We developed a software tool prototype based on the system architecture described above. We included a small subset of classification capabilities – linear SVMs and logistic regression – with a goal to demonstrate, evaluate and visualize differentially private active learning functionality[2]. A screenshot of the prototype during operation is depicted in Fig. 6.

The left pane of the screenshot represents the Input Pane of the prototype. Here, the user is able to specify the location from which the prototype gains access to the training and test (evaluation) datasets. As is typical in data classification, a portion of the training data

[2]We plan to make our tool available upon request for researchers to run their own experiments and expand the set of available classifiers and online active learning strategies.

FIGURE 6. A screenshot of prototype software system showing differentially private active learning on the `KDD Cup 1999` dataset from the UCI machine learning repository.

is extracted and used as a validation dataset. The analyst sees all the attributes of the data and can choose the attributes that he wants to include in the classification algorithm. He can choose the classifier, as well as the mode of operation (output or objective perturbation, active learning, etc.) and the values of parameters such as the regularization and learning rate of the classifier, batch sizes, and the differential privacy parameters, ϵ_s and ϵ_g. Our prototype does not implement (ϵ, δ) differential privacy at this stage.

The right half of the screenshot represents the Output Pane of the prototype, in which the user is able to view the various aspects of classifier performance. The Output Pane displays the classification accuracy plotted against the time, where the time parameter is captured by the checkpoint number. Essentially, at each checkpoint, the tool retrieves the current values of the classification parameters, and checks the anomaly detection performance on the validation dataset and the *next* batch from the training dataset stream. Since, we want to evaluate the performance with and without privacy, the accuracy of a non-private active classifier is also displayed at each checkpoint. After the classifier is trained using the FLMB approach, it is evaluated on a test (evaluation) dataset, and its prediction performance

is depicted both graphically and in a table of relevant metrics including the classification accuracy, precision, recall, specificity, F1 Score and MCC.

5.3. **Evaluation of Prototype.** We experimented with several datasets, but will describe the results obtained with the `KDD Cup 1999` dataset. As mentioned earlier, we extracted training and validation data from the first 400,000 samples. In particular, from those 400,000 samples, we extracted 800 labeled samples uniformly at random with equal number of normal and anomalous samples and used them as our training set. Additionally, we extracted 200 randomly chosen samples, again with equal numbers of normal samples and anomalous samples, and used them as our validation set. Finally, the last 94021 samples were used as our test (evaluation) set. As described in Section 3 the only changes we made to the publicly available dataset was to remove two irrelevant features and convert 7 categorical features into binary vectors, giving a total of 120 features. Note that, unlike the results shown in Fig. 2 which were averaged over a number of simulations, the software tool shows individual runs. Indeed, providing the ability to examine such runs, and to reason about the choice of parameter values for differentially private classification was one of the motivations for building the tool.

Motivated by the evaluation results from Section 3, which showed the superiority of FLMB over FLSW and Bernoulli selection strategy over the exponential selection strategy, the results we discuss here all use the FLMB approach with Bernoulli selection of the informative samples (see Section 3.2). We have shown in earlier work that the online active learning strategy provides superior classification performance than its static non-adaptive counterpart [8]. Here, we will compare the classification performance with and without regard for informativeness under different privacy constraints. Observe in Fig. 7 that for relatively high values of ϵ_s, (i.e., lower privacy), the classification performance is retained irrespective of whether the samples sent to the oracle are highly informative ($\tau = 0.8$) or less informative ($\tau = 0.2$). In contrast, for lower values of ϵ_s, (i.e., stronger privacy), the classification accuracy drops dramatically with respect to the non-private classifier when the samples sent to the oracle are less informative, as depicted in Fig. 7.

6. Related Work

Anomaly detection is a key application area for machine learning techniques. In this paper, in contrast to studying a particular application for anomaly detection, we focus on the privacy issues that may arise in learning an anomaly detector using sensitive streaming data. A 2009 survey by Chandola et al. [10] discusses several approaches for anomaly detection as well as applications. Hodge and Austin [26] survey many outlier detection techniques, some using machine learning methods. Omar et al. [38] provide an overview of machine learning techniques for anomaly detection. More recently, many researchers have proposed employing deep learning for anomaly detection in a variety of applications [13, 29, 56]. We evaluate our method on the `KDD Cup '99` dataset [40], which is a network intrusion detection problem widely studied by the machine learning community (for example, Tang and Cao [51]).

The online learning framework we use has been studied extensively in the machine learning community, starting with the work of Zinkevich [60]. Shalev-Shwartz's 2011 survey [45] gives a comprehensive introduction to the online learning problem; our algorithm is an instance of online learning in that framework. McMahan's 2014 survey [35] covers more recent work on adaptive regularizers. We do not incorporate this type of adaptivity

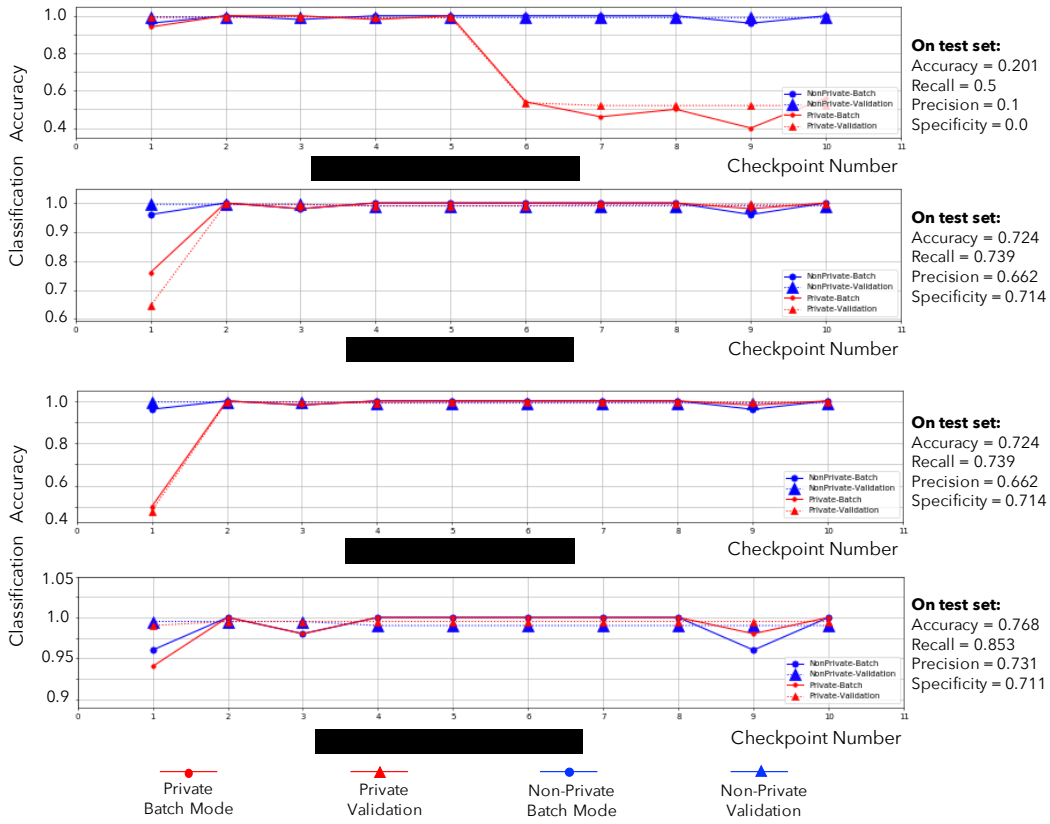


FIGURE 7. Informativeness based sampling with mini-batch updates provides a superior privacy/utility tradeoff. All the above results are obtained with batch size 50, 10 checkpoints, and an SVM classifier with regularization parameter, $\lambda = 0.01$. Results of the final trained classifier on a training set of 94,021 samples are given on the side of each plot.

in our work, but these ideas combined with our approach could be used to yield anomaly detection methods for scenarios in which the data distribution changes over time.

Active learning has received significant attention in the recent years. In active learning, the learning algorithm can adaptively select training examples based on previous data; this adaptivity can lead to significant improvements in sample complexity over "passive learning" approaches that use a random sample of the data [3, 44]. There is a rich body of work on active learning; we refer the reader to the surveys by Settles [44], Fu et al. [22], and the monograph by Hanneke [25] for more details. Recently, deep active learning approaches have been proposed in different applications [21, 23, 33, 46, 59]. In contrast, our approach is based on SVM, a "shallow" learning method.

Our approach uses an active learning heuristic proposed by Tong and Koller [52], which is a simple "pool-based" active learning algorithm using a support vector machine classifier. In pool-based active learning [25], the learning algorithm selects samples to be labeled from a pool of unlabeled data. We propose a "stream-based" version of this heuristic. In stream-based active learning, in each iteration t the algorithm is given point $\mathbf{x}_t$ and has to decide whether or not to request the label y_t from an oracle. Sabato and Hess [42] recently

identified conditions under which stream-based and pool-based active learning can simulate each other. A different line of work in active learning considers the case where the algorithm can choose $\mathbf{x}$ as well. Such active query algorithms [9] are related to probabilistic bisection methods [27].

Differential privacy has been widely studied since its introduction in 2006 [16]; Dwork and Roth's book [19] provides an inclusive review. Duchi et al. [15] study statistical learning problems under local privacy conditions measured by mutual information between private observations and non-private original data, as well as by differential privacy. Dwork et al. formalize the notion of pan-privacy [17, 18] for streaming algorithms. Roughly speaking, pan-private algorithms guarantee differential privacy even when their internal memory state becomes available to an adversary. We partially expose the current state of the algorithm by publishing the learned classifier over time. However, we do assume the buffer of labeled points is kept secret and so do not guarantee pan-privacy.

Online learning has also been studied with differential privacy [2, 28, 53, 58]. Balcan and Feldman [4] describe a framework for designing offline active learning algorithms that are tolerant to classification noise and show that these algorithms are also differentially private. Our algorithms operate in an online manner and use a specific differentially private online active learning algorithm.

7. Discussion

In this paper, we introduced a differentially private learning algorithm and a web tool prototype. We draw some conclusions from each of these contributions.

Differentially Private Online Learning Algorithm: We proposed randomized query strategies for privacy-preserving selection of informative instances in a stream, and explored two mini-batching techniques that improve the classifier performance of the Tong-Koller-based active learning heuristic in our setting. We prove theoretically that the techniques provide differential privacy. When applied in the context of SVM classifiers, the proposed algorithms have acceptable generalization performance while preserving differential privacy of the users whose data is given as input to the algorithms. In particular, mini-batching reduces the variance in the results and improves the convergence of our algorithms. Although the differential privacy guarantees for the mini-batch methods are the same as those for the immediate update method, the worst case scenarios are less likely to happen as the selection window/batch size gets larger. We demonstrated empirically that a time-varying selection strategy where the informativeness criterion becomes stricter with time can substantially reduce label costs while showing acceptable generalization performance, as long as the mini-batching structure is not compromised.

Web Tool Prototype: Differential privacy is an unfamiliar concept for most practitioners of data science: Even with an understanding of the technical definitions, it is difficult to set privacy parameters because – unlike precision, recall, false positive rates, etc. – DP parameters do not have an immediately accessible interpretation. We created a web application prototype that enables analysts to conduct exploratory data analysis with differentially private online learning systems. The prototype uses our privacy-aware active learning heuristics to update classifier models using informative samples from the input stream. It allows the analyst to observe the changes in classifier performance over successive batches of streaming data and relate those changes to the privacy parameters and informativeness

thresholds. We believe that, using the same underlying theoretical approach, we can extend the capabilities of the web application to include other classifier models beyond SVMs and logistic regression.

Outlook and future work: Our proposed online active learning algorithm uses standard techniques to create a differentially private version of the Tong-Koller active learning algorithm. It could be possible to use a different subset selection and update rule to permit a more detailed analysis of the utility (see Section 4). In particular, using a rule similar to that of Malago et al. [34] could permit a version space analysis [5, 6] of the update rule. Making this rule differentially private and adapting the analysis is an interesting topic for future work. In this work we focused on classification tasks: extending the algorithm to other forms of empirical risk minimization seems straightforward. However, extensions of the general approach to nonlinear classifiers or other settings could be interesting and useful for practical scenarios.

Recall that our procedure employed a fixed cost function over time; designing cost-aware algorithms for settings where label cost increases with the number of queries seems challenging. Investigating tradeoffs with respect to ϵ_s, ϵ_g, batch size and classifier learning parameters using a software tool such as the one described herein, could shed light on how stronger privacy guarantees affect the speed of learning and label complexity. Identifying how privacy risk and label complexity interact could yield algorithms which can switch between more and less aggressive active learning techniques.

Finally, because this work is motivated by problems in anomaly detection, evaluating and adapting the private online active learning framework here to domain-specific problems could give more guidance on how to set ϵ in practical scenarios. Implementing practical differentially private machine learning algorithms (as opposed to statistical aggregates/estimators) can tell us when reasonable privacy guarantees are or are not feasible.

Acknowledgments

This material is based upon work supported by the U.S. Department of Homeland Security under Award Number 2009-ST-061-CCI002 and contract HSHQDC-16-A-B0005/HSHQDC-16-J-00371, by the Defense Advanced Research Projects Agency (DARPA) and Space and Naval Warfare Systems Center, Pacific (SSC Pacific) under contract No. N66001-15-C-4070, and by the National Science Foundation under award CCF-1453432.

References

[1] Abadi, Martin, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang (2016) "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, ACM.

[2] Agarwal, Naman and Karan Singh (2017) "The price of differential privacy for online learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 32–40, JMLR. org.

[3] Balcan, Maria-Florina, Alina Beygelzimer, and John Langford (2006) "Agnostic active learning," in *Proceedings of the 23rd international conference on Machine learning*, pp. 65–72, URL: http://doi.acm.org/10.1145/1143844.1143853.

[4] Balcan, Maria-Florina and Vitaly Feldman (2013) "Statistical active learning algorithms," in *Advances in Neural Information Processing Systems*, pp. 1295–1303, URL: http://papers.nips.cc/paper/5101-statistical-active-learning-algorithms.pdf.

[5] Balcan, Maria Florina and Steve Hanneke (2012) "Robust Interactive Learning," in Mannor, Shie, Nathan Srebro, and Robert C. Williamson eds. *Proceedings of the 25th Annual Conference on Learning Theory*, Vol. 23 of Proceedings of Machine Learning Research, pp. 20.1–20.34, Edinburgh, Scotland: PMLR, 25–27 Jun, URL: http://proceedings.mlr.press/v23/balcan12c.html.

[6] Balcan, Maria-Florina, Steve Hanneke, and Jennifer Wortman Vaughan (2010) "The true sample complexity of active learning," *Machine learning*, Vol. 80, No. 2-3, pp. 111–139.

[7] Bassily, Raef, Adam Smith, and Abhradeep Thakurta (2014) "Private Empirical Risk Minimization: Efficient algorithms and tight error bounds," in *2014 IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 464–473, URL: http://dx.doi.org/10.1109/FOCS.2014.56.

[8] Bittner, Daniel, Shantanu Rane, Alejandro Brito, and Rebecca Wright (2018) "A Software Framework for Testing and Evaluation of Differentially Private Active Learning Schemes," in *Workshop on the Theory and Practice of Differential Privacy*, October.

[9] Castro, Rui M. and Robert D. Nowak (2008) "Minimax bounds for active learning," *IEEE Transactions on Information Theory*, Vol. 54, No. 5, pp. 2339–2353, URL: http://dx.doi.org/10.1109/TIT.2008.920189.

[10] Chandola, Varun, Arindam Banerjee, and Vipin Kumar (2009) "Anomaly Detection: A Survey," *ACM computing surveys (CSUR)*, Vol. 41, No. 3, pp. 15:1–15:58, URL: http://doi.acm.org/10.1145/1541880.1541882.

[11] Chaudhuri, Kamalika, Claire Monteleoni, and Anand D. Sarwate (2011) "Differentially Private Empirical Risk Minimization," *Journal of Machine Learning Research*, Vol. 12, pp. 1069–1109, URL: http://www.jmlr.org/papers/v12/chaudhuri11a.

[12] Chaudhuri, Kamalika and Staal A Vinterbo (2013) "A stability-based validation procedure for differentially private machine learning," in *Advances in Neural Information Processing Systems*, pp. 2652–2660.

[13] Du, Min, Feifei Li, Guineng Zheng, and Vivek Srikumar (2017) "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1285–1298, ACM.

[14] Duchi, John C., Michael I. Jordan, and Martin J. Wainwright (2013) "Local Privacy and Statistical Minimax Rates," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 429–438, URL: http://dx.doi.org/10.1109/FOCS.2013.53.

[15] ——— (2014) "Privacy Aware Learning," *Journal of the ACM (JACM)*, Vol. 61, No. 6, pp. 38:1–38:57, URL: http://doi.acm.org/10.1145/2666468.

[16] Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith (2006) "Calibrating Noise to Sensitivity in Private Data Analysis," in *Theory of Cryptography*, URL: http://dx.doi.org/10.1007/11681878_14.

[17] Dwork, Cynthia, Moni Naor, Toniann Pitassi, and Guy N. Rothblum (2010a) "Differential privacy under continual observation," in *Proceedings of the forty-second ACM symposium on Theory of computing*, pp. 715–724, URL: http://doi.acm.org/10.

1145/1806689.1806787.

[18] Dwork, Cynthia, Moni Naor, Toniann Pitassi, Guy N. Rothblum, and Sergey Yekhanin (2010b) "Pan-Private Streaming Algorithms," in *Proceedings of The First Symposium on Innovations in Computer Science (ICS 2010)*, URL: http://www.wisdom.weizmann.ac.il/mathusers/naor/PAPERS/pan_private.pdf.

[19] Dwork, Cynthia and Aaron Roth (2014) "The Algorithmic Foundations of Differential Privacy," *Foundations and Trends in Theoretical Computer Science*, Vol. 9, No. 3-4, pp. 211–407, URL: http://dx.doi.org/10.1561/0400000042.

[20] Feldman, Vitaly, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta (2018) "Privacy amplification by iteration," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 521–532, IEEE.

[21] Feng, Chen, Ming-Yu Liu, Chieh-Chi Kao, and Teng-Yok Lee (2017) "Deep active learning for civil infrastructure defect detection and classification," in *Computing in Civil Engineering 2017*, pp. 298–306.

[22] Fu, Yifan, Xingquan Zhu, and Bin Li (2013) "A survey on instance selection for active learning," *Knowledge and Information Systems*, Vol. 35, No. 2, pp. 249–283, URL: http://dx.doi.org/10.1007/s10115-012-0507-8.

[23] Gal, Yarin, Riashat Islam, and Zoubin Ghahramani (2017) "Deep bayesian active learning with image data," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1183–1192, JMLR. org.

[24] Ghassemi, M., A. Sarwate, and R. Wright (2016) "Differentially private online active learning with applications to anomaly detection," in *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*, pp. 117–128, ACM.

[25] Hanneke, Steve (2011) "Rates of convergence in active learning," *The Annals of Statistics*, Vol. 39, No. 1, pp. 333–361, URL: http://dx.doi.org/10.1214/10-AOS843.

[26] Hodge, Victoria J. and Jim Austin (2004) "A survey of outlier detection methodologies," *Artificial Intelligence Review*, Vol. 22, No. 2, pp. 85–126, URL: http://dx.doi.org/10.1007/s10462-004-4304-y.

[27] Horstein, Michael (1963) "Sequential transmission using noiseless feedback," *IEEE Transactions on Information Theory*, Vol. 9, No. 3, pp. 136–143, URL: http://dx.doi.org/10.1109/TIT.1963.1057832.

[28] Jain, Prateek, Pravesh Kothari, and Abhradeep Thakurta (2012) "Differentially Private Online Learning," in *Proceedings of the 25th Annual Conference on Learning Theory (COLT 2012)*, Vol. 23 of JMLR Workshop and Conference Proceedings, pp. 24.1–24.34, URL: http://www.jmlr.org/proceedings/papers/v23/jain12/jain12.

[29] Javaid, Ahmad, Quamar Niyaz, Weiqing Sun, and Mansoor Alam (2016) "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pp. 21–26.

[30] Ji, Zhanglong, Zachary C Lipton, and Charles Elkan (2014) "Differential privacy and machine learning: a survey and review," *arXiv preprint arXiv:1412.7584*.

[31] Kairouz, Peter, Sewoong Oh, and Pramod Viswanath (2015) "The Composition Theorem for Differential Privacy," in *Proceedings of The 32nd International Conference on Machine Learning*, pp. 1376–1385, URL: http://www.jmlr.org/proceedings/papers/v37/kairouz15.

[32] Kifer, Daniel, Adam Smith, and Abhradeep Thakurta (2012) "Private Convex Empirical Risk Minimization and High-dimensional Regression," in Mannor, Shie,

Nathan Srebro, and Robert C. Williamson eds. *Proceedings of the 25th Annual Conference on Learning Theory (COLT '12)*, Vol. 23 of JMLR Workshop and Conference Proceedings, pp. 25.1–25.40, Edinburgh, Scotland, June, URL: http://jmlr.csail.mit.edu/proceedings/papers/v23/kifer12/kifer12.pdf.

[33] Liu, Peng, Hui Zhang, and Kie B Eom (2017) "Active deep learning for classification of hyperspectral images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 10, No. 2, pp. 712–724.

[34] Malago, Luigi, Nicolo Cesa-Bianchi, and J Renders (2014) "Online active learning with strong and weak annotators," in *NIPS Workshop on Learning from the Wisdom of Crowds*.

[35] McMahan, H. Brendan (2014) "A Survey of Algorithms and Analysis for Adaptive Online Learning," *arXiv preprint arXiv:1403.3465*, URL: http://arxiv.org/abs/1403.3465.

[36] McSherry, Frank (2010) "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," *Communications of the ACM*, Vol. 53, No. 9, pp. 89–97, URL: http://doi.acm.org/10.1145/1810891.1810916.

[37] McSherry, Frank and Kunal Talwar (2007) "Mechanism Design via Differential Privacy," in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 94–103, October, URL: http://dx.doi.org/10.1109/FOCS.2007.41.

[38] Omar, Salima, Asri Ngadi, and Hamid H Jebur (2013) "Machine learning techniques for anomaly detection: an overview," *International Journal of Computer Applications*, Vol. 79, No. 2.

[39] Quionero-Candela, Joaquin, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence (2009) *Dataset Shift in Machine Learning*, Cambridge, MA: MIT Press.

[40] Rosset, Saharon and Aron Inger (2000) "KDD-cup 99: knowledge discovery in a charitable organization's donor database," *ACM SIGKDD Explorations Newsletter*, Vol. 1, No. 2, pp. 85–90, URL: http://doi.acm.org/10.1145/846183.846204, DOI: 10.1145/846183.846204.

[41] Rubinstein, Benjamin I. P., Peter L. Bartlett, Ling Huang, and Nina Taft (2012) "Learning in a Large Function Space: Privacy-Preserving Mechanisms for SVM Learning," *Journal of Privacy and Confidentiality*, Vol. 4, No. 1, pp. 65–100, URL: http://repository.cmu.edu/jpc/vol4/iss1/4/.

[42] Sabato, Sivan and Tom Hess (2016) "Interactive algorithms: from pool to stream," in *Proceedings of the 2016 Conference On Learning Theory (COLT 2016)*, pp. 1419–1439, URL: http://www.jmlr.org/proceedings/papers/v49/sabato16.

[43] Sarwate, Anand D and Kamalika Chaudhuri (2013) "Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data," *IEEE signal processing magazine*, Vol. 30, No. 5, pp. 86–94.

[44] Settles, Burr (2010) "Active learning literature survey. Computer Sciences Technical Report 1648," URL: http://burrsettles.com/pub/settles.activelearning.pdf.

[45] Shalev-Shwartz, Shai (2011) "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, Vol. 4, No. 2, pp. 107–194, URL: http://dx.doi.org/10.1561/2200000018.

[46] Shen, Yanyao, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar (2017) "Deep active learning for named entity recognition," *arXiv preprint arXiv:1707.05928*.

[47] Shokri, Reza and Vitaly Shmatikov (2015) "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1310–1321.

[48] Song, Shuang, Kamalika Chaudhuri, and Anand Sarwate (2015a) "Learning from data with heterogeneous noise using sgd," in *Artificial Intelligence and Statistics*, pp. 894–902.

[49] Song, Shuang, Kamalika Chaudhuri, and Anand D. Sarwate (2013) "Stochastic Gradient Descent with Differentially Private Updates," in *Proceedings of the 2013 Global Conference on Signal and Information Processing (GlobalSIP 2013)*, pp. 245–248, URL: http://dx.doi.org/10.1109/GlobalSIP.2013.6736861.

[50] ——— (2015b) "Learning from Data with Heterogeneous Noise using SGD," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pp. 894–902, URL: http://jmlr.org/proceedings/papers/v38/song15.html.

[51] Tang, Hua and Zhuolin Cao (2009) "Machine learning-based intrusion detection algorithms," *Journal of Computational Information Systems*, Vol. 5, No. 6, pp. 1825–1831.

[52] Tong, Simon and Daphne Koller (2001) "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, Vol. 2, pp. 45–66, URL: http://www.jmlr.org/papers/v2/tong01a.html.

[53] Wang, Yining and Anima Anandkumar (2016) "Online and differentially-private tensor decomposition," in *Advances in Neural Information Processing Systems*, pp. 3531–3539.

[54] Warner, Stanley L. (1965) "Randomized response: a survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, Vol. 60, No. 309, pp. 63–69, URL: http://dx.doi.org/10.1080/01621459.1965.10480775.

[55] Wasserman, Larry and Shuheng Zhou (2010) "A Statistical Framework for Differential Privacy," *Journal of the American Statistical Association*, Vol. 105, No. 489, pp. 375–389, URL: http://dx.doi.org/10.1198/jasa.2009.tm08651.

[56] Xu, Dan, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe (2015) "Learning deep representations of appearance and motion for anomalous event detection," *arXiv preprint arXiv:1510.01553*.

[57] Zhang, Jun, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Marianne Winslett (2012) "Functional Mechanism: Regression Analysis under Differential Privacy," *Proceedings of the VLDB Endowment*, Vol. 5, No. 11, pp. 1364–1375, URL: http://vldb.org/pvldb/vol5/p1364_junzhang_vldb2012.pdf.

[58] Zhou, Pan, Yingxue Zhou, Dapeng Wu, and Hai Jin (2016) "Differentially private online learning for cloud-based video recommendation with multimedia big data in social networks," *IEEE transactions on multimedia*, Vol. 18, No. 6, pp. 1217–1229.

[59] Zhou, Shusen, Qingcai Chen, and Xiaolong Wang (2010) "Active deep networks for semi-supervised sentiment classification," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 1515–1523, Association for Computational Linguistics.

[60] Zinkevich, Martin (2003) "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pp. 928–936, URL: http://www.aaai.org/Papers/ICML/2003/ICML03-120.pdf.

Appendix A. Appendices

A.1.

Proof. Consider two databases $\mathcal{D}_x$ and $\mathcal{D}'_x$ that differ in a single point: for some j, the jth point of $\mathcal{D}_x$ is $\mathbf{x}_p$ and the jth point of $\mathcal{D}'_x$ is $\mathbf{x}_q$. The selection procedure reveals $\mathbf{w}_t$ at each iteration t. Since the samples are presented online, until time τ the iterations have the same distribution.

Then,

$$\begin{aligned}\left|\log \frac{\mathbb{P}\left(s_t=1|\mathcal{D},\mathbf{w}_t\right)}{\mathbb{P}\left(s_t=1|\mathcal{D}',\mathbf{w}_t\right)}\right| &\leq \log \frac{\mathbb{P}\left(s_t=1|i_t=p,\mathbf{w}_t\right)}{\mathbb{P}\left(s_t=1|i_t=q,\mathbf{w}_t\right)}\\ &\leq \log \frac{\mathbb{P}\left(s_t=1|c_p(t)>\tau\right)}{\mathbb{P}\left(s_t=1|c_q(t)<\tau\right)}\\ &\leq \log \frac{p}{1-p}. \end{aligned} \tag{A.1}$$

Similarly, the same result can be shown for the case $s_t = 0$. This concludes the proof. □

A.2.

Proof. Let $\mathbf{x}_p$ and $\mathbf{x}_q$ be the jth entries of two neighboring datasets $\mathcal{D}_x$ and $\mathcal{D}'_x$, respectively. Then,

$$\begin{aligned}&\left|\log \frac{\mathbb{P}\left(s_t=1|\mathcal{D},\mathbf{w}_t\right)}{\mathbb{P}\left(s_t=1|\mathcal{D}',\mathbf{w}_t\right)}\right|\\ &\leq \left|\log \frac{\exp\left(-\max\{b,d(\mathbf{x}_p,\mathbf{w}_t)\}\epsilon_s/\Delta\right)}{\exp\left(-\max\{b,d(\mathbf{x}_q,\mathbf{w}_t)\}\epsilon_s/\Delta\right)}\right|\\ &\leq \left|\max\{b,d(\mathbf{x}_q,\mathbf{w}_t)\}-\max\{b,d(\mathbf{x}_p,\mathbf{w}_t)\}\right|\frac{\epsilon_{\mathsf{exp}}}{\Delta}\\ &\leq (M-b)\frac{\epsilon_{\mathsf{exp}}}{\Delta}\\ &= \epsilon_{\mathsf{exp}}. \end{aligned} \tag{A.2}$$

Similarly, considering the assumption $\exp(-b\epsilon_{\mathsf{exp}}/\Delta) \leq 1/2$, we have

$$\begin{aligned}\left|\log \frac{\mathbb{P}\left(s_t=0|\mathcal{D},\mathbf{w}_t\right)}{\mathbb{P}\left(s_t=0|\mathcal{D}',\mathbf{w}_t\right)}\right| &\leq \log \frac{1-\exp(-M\epsilon_{\mathsf{exp}}/\Delta)}{1-\exp(-b\epsilon_{\mathsf{exp}}/\Delta)}\\ &\leq \epsilon_{\mathsf{exp}}. \end{aligned} \tag{A.3}$$

This concludes the proof. □

A.3.

Proof. let $\mathbf{x}_p$ be the jth entry of $\mathcal{D}_x$ and $\mathbf{x}_q$ be the jth entry of $\mathcal{D}_x$. Furthermore, assume without loss of generality that $|\langle \mathbf{w}_t, \mathbf{x}_p \rangle| \leq |\langle \mathbf{w}_t, \mathbf{x}_q \rangle|$.

$$\begin{aligned}
&\frac{\mathbb{P}\left(\mathbf{w}'|\mathcal{D}, \mathbf{w}\right)}{\mathbb{P}\left(\mathbf{w}'|\mathcal{D}', \mathbf{w}\right)} \\
&\leq \frac{\gamma e^{-\frac{\epsilon_g B}{2\eta M}\|\mathbf{w}'-\mathbf{w}+\eta\lambda\mathbf{w}-\frac{1}{B}\sum_{i\in\mathcal{B}:i\neq p}\eta y_i \mathbf{x}_i u_i - \frac{\eta}{B} y_p \mathbf{x}_p u_p\|}}{\gamma e^{-\frac{\epsilon_g B}{2\eta M}\|\mathbf{w}'-\mathbf{w}+\eta\lambda\mathbf{w}-\frac{1}{B}\sum_{i\in\mathcal{B}:i\neq q}\eta y_i \mathbf{x}_i u_i - \frac{\eta}{B} y_q \mathbf{x}_q u_q\|}} \\
&\leq e^{(\epsilon_g/2M)\|\mathbf{x}_p - \mathbf{x}_q\|} \\
&\leq e^{\epsilon_g}. \qquad \text{(A.4)}
\end{aligned}$$

Similarly, we can show that if $B = \emptyset$,

$$\mathbb{P}\left(\mathbf{w}'|\mathcal{D}, \mathbf{w}\right) = \mathbb{P}\left(\mathbf{w}'|\mathcal{D}', \mathbf{w}\right)$$

and thus

$$\left|\log \frac{\mathbb{P}\left(\mathbf{w}'|\mathcal{D}, \mathbf{w}\right)}{\mathbb{P}\left(\mathbf{w}'|\mathcal{D}', \mathbf{w}\right)}\right| = 0 \leq \epsilon_g. \qquad \text{(A.5)}$$

so the result is differentially private for two neighboring datasets. This concludes the proof. □

A.4.

Proof. Each selection step incurs ϵ_s privacy risk. If we show that each update incurs at most ϵ_g privacy risk, so the risk per sample (by composition) is at most $\epsilon_s + \epsilon_g$.

Suppose the algorithm runs for T iterations and define the matrix $W_T = [\mathbf{w}_0^\top, \mathbf{w}_1^\top, \mathbf{w}_2^\top, \ldots, \mathbf{w}_T^\top]^\top$ of revealed values. From this an adversary can infer $Q_T = [s_1, s_2, \ldots, s_T]$, the vector of selection results. Let $\mathcal{D}$ and $\mathcal{D}'$ be two neighboring databases that differ in the jth element: $\mathbf{x}_j = \mathbf{x}_p$ in $\mathcal{D}$ and $\mathbf{x}'_j = \mathbf{x}_q$ in $\mathcal{D}'$. We calculate the log-likelihood ratio:

$$\begin{aligned}
&\left|\log \frac{\mathbb{P}(W_T, Q_T|\mathcal{D})}{\mathbb{P}(W_T, Q_T|\mathcal{D}')}\right| \\
&= \left|\log \frac{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t, s_t|\mathcal{D}, W_{t-1})}{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t, s_t|\mathcal{D}', W_{t-1})}\right| \\
&= \left|\log \frac{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t|\mathcal{D}, W_{t-1}, s_t)\mathbb{P}(s_t|\mathcal{D}, W_{t-1})}{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t|\mathcal{D}', W_{t-1}, s_t)\mathbb{P}(s_t|\mathcal{D}', W_{t-1})}\right|. \qquad \text{(A.6)}
\end{aligned}$$

Now, we use the fact that up until time j, the distribution of the two algorithms' decisions are identical:

$$\begin{aligned}
&\left|\log \frac{\mathbb{P}(W_T, Q_T|\mathcal{D})}{\mathbb{P}(W_T, Q_T|\mathcal{D}')}\right| \\
&\overset{(a)}{=} \left|\log \frac{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t|\mathcal{D}, \mathbf{w}_{t-1}, s_t)\mathbb{P}(s_t|\mathcal{D}, \mathbf{w}_{t-1})}{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t|\mathcal{D}', \mathbf{w}_{t-1}, s_t)\mathbb{P}(s_t|\mathcal{D}', \mathbf{w}_{t-1})}\right| \\
&\overset{(b)}{=} \left|\log \frac{\mathbb{P}(\mathbf{w}_j|\mathcal{D}, \mathbf{w}_{j-1}, s_j)\mathbb{P}(s_j|\mathcal{D}, \mathbf{w}_{j-1})}{\mathbb{P}(\mathbf{w}_j|\mathcal{D}', \mathbf{w}_{j-1}, s_j)\mathbb{P}(s_j|\mathcal{D}', \mathbf{w}_{j-1})}\right| \\
&\leq \left|\log \frac{\mathbb{P}(\mathbf{w}_j|\mathcal{D}, \mathbf{w}_{j-1}, s_j)}{\mathbb{P}(\mathbf{w}_j|\mathcal{D}', \mathbf{w}_{j-1}, s_j)}\right| + \left|\log \frac{\mathbb{P}(s_j|\mathcal{D}, \mathbf{w}_{j-1})}{\mathbb{P}(s_j|\mathcal{D}', \mathbf{w}_{j-1})}\right| \\
&\leq \epsilon_g + \epsilon_s, \qquad \text{(A.7)}
\end{aligned}$$

Equality (a) above results from the fact that given the most recent classifier $\mathbf{w}_t$, the probability distributions of $\mathbf{w}_t$ and s_t are independent of the previous classifiers. Equality (b) follows from the assumption the observed samples do not appear again in the stream, so except for the iteration where the streams are different, the conditional probabilities are identical. □

A.5.

Proof. Suppose that we run the algorithm for K iterations each consisting of N sample observations and a gradient update. Similar to Theorem 4.5, define the collection of updates $W_K = (\mathbf{w}_0, \mathbf{w}_N, \mathbf{w}_{2N} \ldots, \mathbf{w}_{KN})$ and also $Q_K = (S_1, S_2, \ldots, S_K)$ where S_k indicates if any samples have been labeled during the kth interval. We have

$$\begin{aligned}
&\left|\log \frac{\mathbb{P}(W_{KN}, Q_K|\mathcal{D})}{\mathbb{P}(W_{KN}, Q_K|\mathcal{D}')}\right| \\
&= \left|\log \frac{\prod_{k=1}^{K} \mathbb{P}(\mathbf{w}_{kN}, Q_k|\mathcal{D}, W_{(k-1)N})}{\prod_{k=1}^{K} \mathbb{P}(\mathbf{w}_{kN}, Q_k|\mathcal{D}', W_{(k-1)N})}\right| \\
&\overset{(a)}{=} \left|\log \frac{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_{kN}|\mathcal{D}, \mathbf{w}_{(k-1)N}, s_t)\mathbb{P}(s_t|\mathcal{D}, \mathbf{w}_{(k-1)N})}{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_{kN}|\mathcal{D}', \mathbf{w}_{(k-1)N}, s_t)\mathbb{P}(s_t|\mathcal{D}', \mathbf{w}_{(k-1)N})}\right| \\
&\overset{(b)}{=} \left|\log \frac{\mathbb{P}(\mathbf{w}_j|\mathcal{D}, \mathbf{w}_{j-1}, s_j)\mathbb{P}(s_j|\mathcal{D}, \mathbf{w}_{j-1})}{\mathbb{P}(\mathbf{w}_j|\mathcal{D}', \mathbf{w}_{j-1}, s_j)\mathbb{P}(s_j|\mathcal{D}', \mathbf{w}_{j-1})}\right| \\
&\leq \left|\log \frac{\mathbb{P}(\mathbf{w}_j|\mathcal{D}, \mathbf{w}_{j-1}, s_j)}{\mathbb{P}(\mathbf{w}_j|\mathcal{D}', \mathbf{w}_{j-1}, s_j)}\right| + \left|\log \frac{\mathbb{P}(s_j|\mathcal{D}, \mathbf{w}_{j-1})}{\mathbb{P}(s_j|\mathcal{D}', \mathbf{w}_{j-1})}\right| \\
&\leq \epsilon_s + \epsilon_g, \qquad \text{(A.8)}
\end{aligned}$$

where j here indicates the window in which the jth entries of the datasets appear. Equalities (a) and (b) above are explained in Theorem 4.5. □

A.6. Suppose that we run the algorithm for T iterations each consisting of T_l sample observations and a (possible) gradient update. As before, define the sets of outputs $W_T = (\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_T)$ and $Q_T = (T_1, T_2, \ldots, T_T)$. We have

$$\begin{aligned}
&\left|\log \frac{\mathbb{P}(W_T, Q_T|\mathcal{D})}{\mathbb{P}(W_T, Q_T|\mathcal{D}')}\right| \\
&= \left|\log \frac{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t, T_t|\mathcal{D}, W_{t-1})}{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t, T_t|\mathcal{D}', W_{t-1})}\right| \\
&\overset{(a)}{=} \left|\log \frac{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t|\mathcal{D}, \mathbf{w}_{t-1}, T_t)\mathbb{P}(T_t|\mathcal{D}, \mathbf{w}_{t-1})}{\prod_{t=1}^{T} \mathbb{P}(\mathbf{w}_t|\mathcal{D}', \mathbf{w}_{t-1}, T_t)\mathbb{P}(T_t|\mathcal{D}', \mathbf{w}_{t-1})}\right| \\
&\overset{(b)}{=} \left|\log \frac{\mathbb{P}(\mathbf{w}_j|\mathcal{D}, \mathbf{w}_{j-1}, T_j)\mathbb{P}(T_j|\mathcal{D}, \mathbf{w}_{j-1})}{\mathbb{P}(\mathbf{w}_j|\mathcal{D}', \mathbf{w}_{j-1}, T_j)\mathbb{P}(T_j|\mathcal{D}', \mathbf{w}_{j-1})}\right| \\
&\leq \left|\log \frac{\mathbb{P}(\mathbf{w}_j|\mathcal{D}, \mathbf{w}_{j-1}, T_j)}{\mathbb{P}(\mathbf{w}_j|\mathcal{D}', \mathbf{w}_{j-1}, T_j)}\right| + \left|\log \frac{\mathbb{P}(T_j|\mathcal{D}, \mathbf{w}_{j-1})}{\mathbb{P}(T_j|\mathcal{D}', \mathbf{w}_{j-1})}\right| \\
&\leq \epsilon_s + \epsilon_g. \qquad \text{(A.9)}
\end{aligned}$$

where j indicates the window in which the jth entries of the datasets appear. Equalities (a) and (b) above are explained in Theorem 4.5. □