Re-identification Attack to Privacy-Preserving Data Analysis with Noisy Sample-Mean

Du Su, Hieu Tri Huynh, Ziao Chen, Yi Lu
University of Illinois at Urbana-Champaign
{dusu3,hthuynh2,zchen149,yilu4}@illinois.edu

Wenmiao Lu
Verizon Media
wenl@verizonmedia.com

ABSTRACT

In mining sensitive databases, access to sensitive class attributes of individual records is often prohibited by enforcing field-level security, while only aggregate class-specific statistics are allowed to be released. We consider a common privacy-preserving data analytics scenario where only a noisy sample mean of the class of interest can be queried. Such practice is widely found in medical research and business analytics settings.

This paper studies the hazard of re-identification of entire class caused by revealing a noisy sample mean of the class. With a novel formulation of the re-identification attack as a generalized positive-unlabeled learning problem, we prove that the risk function of the re-identification problem is closely related to that of learning with complete data. We demonstrate that with a one-sided noisy sample mean, an effective re-identification attack can be devised with existing PU learning algorithms. We then propose a novel algorithm, growPU, that exploits the unique property of sample mean and consistently outperforms existing PU learning algorithms on the re-identification task. GrowPU achieves re-identification accuracy of 93.6% on the MNIST dataset and 88.1% on an online behavioral dataset with noiseless sample mean. With noise that guarantees 0.01-differential privacy, growPU achieves 91.9% on the MNIST dataset and 84.6% on the online behavioral dataset.

CCS CONCEPTS

• **Security and privacy → Privacy protections**; • **Computing methodologies → Semi-supervised learning settings**.

KEYWORDS

re-identification; data privacy; positive-unlabeled learning

ACM Reference Format:
Du Su, Hieu Tri Huynh, Ziao Chen, Yi Lu and Wenmiao Lu. 2020. Re-identification Attack to Privacy-Preserving Data Analysis with Noisy Sample-Mean. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining USB Stick (KDD '20), August 23–27, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403148

KDD '20, August 23–27, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403148

1 INTRODUCTION

Privacy is one of the top concerns shared by data miners, data responders, data curators, and regulatory agencies [12, 17, 23]. To that end, sensitive class attributes of user records are protected from individual-level access by enforcing field-level security. Instead, only the aggregate class-specific statistics are allowed to be released. In addition, differential privacy [1, 8, 9, 19, 20] policies are often applied to allow only noisy computations of aggregate statistics (such as the class' mean, variance, etc.)

In this paper, we study a commonly encountered data analytics scenario: the access to the class-label field of each record is prohibited, while only the noisy sample mean of one class is allowed to be queried. Formally, there are two classes C_0 and C_1. A record R_i has the confidential class label $L_i \in C_0, C_1$, and a set of n non-confidential feature fields $F_i = f_0, \ldots, f_n$. For the purpose of privacy preservation, no query is allowed on the confidential class labels. Instead, a noisy sample mean of the records in the class C_0 is available, i.e., $S_0 = \sum_i F_i/N + \delta$, where N is the number of records in the class C_0, and δ is Laplacian noise. Variations of the scenario include only having access to a subset of records and/or taking the noisy sample mean of only a random subset of the class C_0.

Such scenario is commonly encountered in various medical research and business analytics settings, where the labels of individual records, such as positive test results or credit approval results, are not allowed to be released. Nevertheless, in these settings, the aggregate statistics, in particular the class-level sample mean, is considered compliant with privacy requirement, and is allowed to be released. For example, while an internal analyst at a credit card company is not allowed to see the approval status of individual applications, it is common practice for the analyst to query the average income level or the average of recent credit inquires of approved applications.

The class-level aggregate statistics obtained in the aforementioned scenario are considered privacy-preserving, as these records usually have high dimensionality and sparse features. Due to the curse-of-dimensionality, all records are far away from the sample mean, that is, locating records close to the sample mean does not yield useful class information.

In this paper, we study this seemingly innocuous data analysis scenario and demonstrate that *with a one-sided noisy sample mean, a highly effective re-identification adversarial attack can be devised to recover the confidential class labels.* The re-identification attack is established based on a novel formulation of the label recovery task as a generalized positive-unlabeled classification problem [6, 7, 10, 14, 27], where the positive class has the sample mean as the only sample. We also propose a novel algorithm, growPU, that exploits the unique property of sample mean in the learning process to further improve the accuracy. By experimenting on benchmark

datasets and an online user behavioral dataset, the one-sided sample mean has surprisingly powerful ability to reliably re-identify the confidential class labels.

Our main contributions are as follows:

(1) To the best of our knowledge, we are the first to reveal the vulnerability of the high-accuracy re-identification attack based on aggregate statistics of sensitive data;
(2) We are the first to formulate the re-identification attack problem as a generalized positive-unlabeled learning (PU learning) problem and formally establish the relationship between re-identification using a one-sided noisy sample mean and the problem of learning with complete data;
(3) We demonstrate that with a one-sided noisy sample mean, an effective re-identification attack can be devised with existing PU learning algorithms.
(4) We propose a novel algorithm, growPU, that exploits the unique property of sample mean and consistently outperforms the existing PU leaning algorithms on the re-identification task. GrowPU achieves re-identification accuracy of 93.6% on the MNIST dataset and 88.1% on an online behavioral dataset with noiseless sample mean. With noise that guarantees 0.01-differential privacy, growPU achieves 91.9% on the MNIST dataset and 84.6% on the online behavioral dataset.

Organization: We discuss related work in Section 2, and analyze an online user behavioral dataset to expose the neglect of the risk of a re-identification attack in Section 3. We formulate the re-identification problem as a PU Learning problem and prove its relationship to learning with complete data in Section 4. We present the growPU algorithm in Section 5 and experiment results in Section 6.

2 RELATED WORK

Differential Privacy [1, 8, 9, 19, 20]: The purpose of differential privacy is to guard against aggregate query attacks on a database of private records. The basic idea is to add noise to query results so that individual data cannot be deduced from a sequence of queries. The problem we are dealing with is different. Here *only one* query is allowed and a noisy sample mean is returned on one of the classes. The challenge is to devise an effective re-identification attack to recover the confidential labels for all records.

Positive-Unlabeled Learning (PU Learning) [6, 7, 10, 14, 27]: Given positive and negative data, with only a subset of the positive data labeled, PU learning tries to classify all unlabeled data. It is a one-sided problem as only a subset of the *positive* data have labels. In this aspect, it is the closest to our problem as we try to re-identify the confidential labels with only a one-sided noisy sample mean. However, PU learning typically requires sufficient positive data, usually in the order of *hundreds*, to approximate the distribution of the positive class well [14]. Hence it is rather surprising that high accuracy is achieved when we use PU Learning algorithms for re-identification, where only one noisy sample mean is available. We formulate the re-identification problem as a generalized PU learning problem and show its relationship to the problem of learning from complete data.

Semi-supervised Learning [4, 13, 15, 16, 18, 28]: Unlike PU Learning, the semi-supervised learning problem is to classify two or more classes of data, where each class has a subset of data with labels. Hence, it is a two-sided or multi-sided problem. Having labeled data in all classes makes the problem significantly easier, and semi-supervised learning typically achieves better accuracy than PU learning. The re-identification problem is one-sided. However, our growPU algorithm is able to achieve a high accuracy of above 90%, with only a one-sided noisy sample mean, which is comparable to that of the semi-supervised learning scenario.

Unsupervised Learning [2, 11, 22, 24–26]: When no labels are provided, data are classified into a given number of classes based on features of the data alone. Typically intra-cluster distances are minimized and inter-cluster distances are maximized. The challenge is to select an optimal feature space to compute the distance. Since we only have a one-sided noisy sample mean, the information we possess seems only slightly more than that of unsupervised learning. Hence it is surprising that our growPU algorithm is able to achieve $88 - 94\%$ accuracy when clustering algorithms, such as K-means, achieves only $60 - 68\%$ accuracy, depending on the datasets. Note that a random guess will yield 50% accuracy on a balanced dataset.

3 WHY RE-IDENTIFICATION IS NOT PERCEIVED AS A RISK

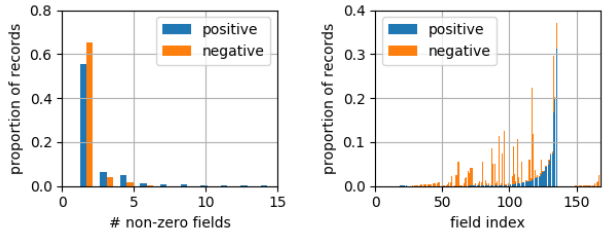
It is commonly considered safe to release aggregate statistics of sensitive data, which are useful for business and research purposes. For example, while an internal analyst at a credit card company is not allowed to see the approval status of individual applications, it is common practice for the analyst to query the average income level or the average of recent credit inquires of approved applications.

Re-identification based on the aggregate data, such as sample mean, is considered ineffective due to certain data characteristics such as extreme feature sparsity, significant feature overlap, curse-of-dimensionality and interspersed clusters. These characteristics are illustrated in this section with an online user behavioral data set from a large internet company. This data set consists of user records sampled from two advertising behavioral segments, which we will refer to as the *positive* and *negative* class respectively. Each of the two classes contains 10,000 sampled user records. Each user record contains 168 categorical variables, which correspond to a variety of online user behaviors.

3.1 High Dimensionality and Extreme Sparsity

The 168 fields in our data set are extremely sparsely populated. Figure 1(a) shows that 72.3% of positive records and 55.3% of negative records have *only 1 none-zero value* out of all 168 fields. In addition, 95 % of positive records have at most 3 non-zero values, and 95% of negative records have at most 6 non-zero values.

High dimensionality and extreme sparsity are typically observed in business analytics data from NoSQL databases [3, 21] where there is a large variety of different user behaviors and each user only takes a very small set of actions. As a result, the database contains a large number of columns, but each record is very sparse. We shall see in Section 3.3 that the high dimensionality and extreme sparsity make naive re-identification ineffective, and is an important reason why people regard the release of aggregate statistics safe.



(a) Proportion of records against number of non-zero fields. Records are extremely sparse

(b) Proportion of records with non-zero value on each field, Most non-zero fields of the two classes overlap

Figure 1: Visualization of the online behavioral dataset

3.2 Feature Overlap

The first attempt at re-identification is to identify the unique fields of the two classes. However, despite the sparsity, the non-zero fields of the two classes cover 154 out of 168 fields, and *most non-zero fields of the two classes overlap*. Figure 1(b) plots the percentage of records with non-zero value in a given field. The 168 fields are plotted on the horizontal axis, divided into four groups: Position 1-14 are fields that no data has non-zero values; Position 15-24 are fields where only the positive class has non-zero values; Position 25-136 are fields where the two classes overlap, sorted by the percentage of the positive class having non-zero value in a given field; Position 137 - 168 are fields where only the negative class has non-zero values.

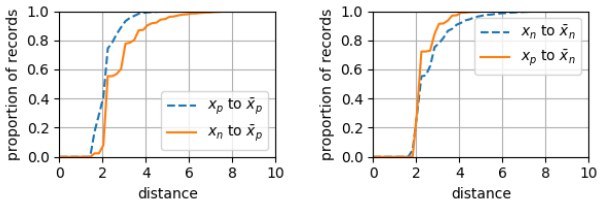
The large overlap between the positive and negative classes makes it impossible to re-identify the samples based on the unique fields of the two classes. Out of a total of 168 fields, only 10 fields are unique to the positive class, and 32 unique to the negative class. Moreover, *only 73* positive samples have non-zero value on the 10 fields unique to the positive class, and *only 682* negative samples have non-zero value on the 32 fields unique to the negative class. The majority of samples, 99.4% of the positive class and 93.2% of the negative class, have non-zero values *only on the shared fields*.

3.3 Curse of Dimensionality

Next we consider a re-identification attack using the sample mean of the positive class, which we will refer to as P-mean. Analogously we refer to the sample mean of the negative class as N-mean. A naive attempt is to select the samples close to the P-mean and label them positive.

Figure 2(a) shows the distribution of the respective distances of the positive and negative samples from the P-mean. We observe that the two distributions are not well-separated. First, there are no positive samples within a distance of 1.4 from the P-mean. This is due to the curse of dimensionality, which implies that any two samples are far apart in a high-dimensional space. As a result, the positive samples are far from each other, and also far from their mean. Secondly, over 90% of *negative* samples are within a distance of 4.0 from the P-mean, i.e., the negative samples are *not further* from the P-mean than the positive samples. In addition, the average distance between the positive samples and the P-mean is 2.14, and the average distance between the negative samples and the P-mean is 2.78. We observe that the difference between the two average distances is small. Together the above statistics show that the distances of positive and negative samples from the P-mean are not sufficient to separate the two classes.

Figure 2(b) shows the distribution of the respective distances of the positive and negative samples from the N-mean. The distributions are even less separable than Fig. 2(a). In fact, up to the 50-th percentile, the positive and negative samples have the same distance distribution! That is, among the samples closest to the N-mean, the positive and negative samples are absolutely indistinguishable based on distance. The average distance of the positive samples to the N-mean is 2.36, and the average distance of negative samples to the N-mean is 2.62. We observe that, on average, the negative samples are *even further away* from the N-mean than the positive samples. It is hence not surprising that people regard the release of aggregate statistics safe.



(a) CDF of Sample Distance to Positive Sample Mean

(b) CDF of Sample Distance to Negative Sample Mean

Figure 2: The two classes cannot be separated based on their respective distances to P-mean or N-mean.

3.4 Interspersed Clusters

A more sophisticated attempt at re-identification is to use existing clustering methods. The K-means clustering algorithm yields an accuracy of 60%. Note a random guess will yield 50% accuracy on a balanced dataset.

The poor performance of the K-means clustering algorithm is potentially due to the clusters being interspersed with each other. Figure 3 shows the first two dimensions of the PCA decomposition for positive and negative data. There are 4 clusters of negative samples centered around $(-1, -1)$, $(-1, 1)$, $(1, -1)$ and $(1, 1)$, and 4 clusters of positive samples centered around $(-1.5, 0.5)$, $(-1, 1)$, $(0, 0)$ and $(1, 1.5)$. The nearest neighbor for each cluster is of the opposite class, and the distance between two clusters of the same class can be large.

4 RE-IDENTIFICATION ATTACK AS PU LEARNING

We formulate the re-identification attack problem as a PU Learning problem in this section. We start with a review of notations in 4.1, and introduce the PU learning problem in 4.2. We present the formulation in 4.3 together with proofs establishing the formal relationship between the re-identification problem and PU learning.

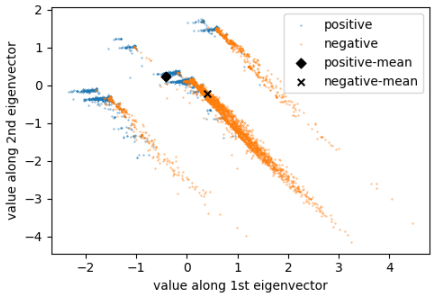


Figure 3: The first two PCA dimensions of positive and negative samples show interspersed clusters.

4.1 Preliminary

Let random variables $X \in \mathbb{R}^d$ and $Y \in \{-1, 1\}$ denote a sample and its label. Let $P_{XY}(x, y)$ be the joint distribution of the sample-label pair (X, Y), and $P_X(x)$ and $P_Y(y)$ be marginal distribution for samples and labels. We denote the marginal distributions of positive samples and negative samples by $P_p(x) = \mathcal{P}(X = x|Y = 1)$ and $P_n(x) = \mathcal{P}(X = x|Y = -1)$ respectively. Let $\pi_p = P_Y(1)$ and $\pi_n = P_Y(-1)$ denote the probability mass for positive and negative classes.

Let $l(x, y) : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ be a non-negative risk function. The expected risk of a decision function g over $P_{XY}(x, y)$ is:

$$\begin{aligned} R(g) &= \mathbb{E}[l(g(x), y)] \\ &= \pi_p \mathbb{E}_p[l(g(x), 1)] + \pi_n \mathbb{E}_n l[(g(x), -1)], \end{aligned}$$

where $\mathbb{E}_p$ and $\mathbb{E}_n$ are expectations taken over P_p and P_n respectively.

Suppose we are given a family of decision functions $\mathcal{G} = \{g : \mathbb{R}^d \to \mathbb{R}\}$, we aim to find the optimal

$$g^* = \arg\min_{g \in \mathcal{G}} R(g).$$

In practice, the true distribution P_{XY} is usually not available. We can only access a dataset drawn from the true distribution. Denote independently sampled positive and negative samples by $\mathcal{X}_p = \{x : x \sim P_p(x)\}$ and $\mathcal{X}_n = \{x : x \sim P_n(x)\}$. Let $n_p = \left\|\mathcal{X}_p\right\|_0$ and $n_n = \left\|\mathcal{X}_n\right\|_0$ denote the number of positive and negative samples respectively. We estimate π_p and π_n by $\frac{\pi_p}{\pi_n} = \frac{n_p}{n_n}$. We define the empirical risk

$$\hat{R}(g) = \frac{\pi_p}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1) + \frac{\pi_n}{n_n} \sum_{x \in \mathcal{X}_n} l(g(x), -1), \tag{1}$$

and find the optimal empirical decision function

$$\hat{g} = \arg\min_{g \in \mathcal{G}} \hat{R}(g).$$

4.2 PU Learning

The PU learning problem only has a subset of the positive samples labeled. We can access two independently sampled sets $\tilde{\mathcal{X}}_p = \{x : x \sim P_p(x)\}$ and $\tilde{\mathcal{X}}_u = \{x : x \sim P_X(x)\}$, which are the positive labeled set and the unlabeled set respectively. And the proportion of positive samples, π_p is given. The objective is to learn a decision function $\hat{g}_{PU}$ to maximize the expected risk $R(g)$.

Du Plessis et al. [7] showed that if the loss function satisfies the symmetric condition:

$$l(g(x), 1) + l(g(x), -1) = 1,$$

we can approximate the expected loss $R(g)$ with positive and unlabeled samples:

$$\begin{aligned} R(g) &= \pi_p \mathbb{E}_p[l(g(x), 1)] + \mathbb{E}_n l[(g(x), -1)] \\ &= 2\pi_p \mathbb{E}_p[l(g(x), 1)] + \mathbb{E}_u l[(g(x), -1)] - \pi_p, \end{aligned}$$

where $\mathbb{E}_u$ is expectation taken over the sample distribution P_X.

We approximate $R(g)$ using empirical data $\tilde{\mathcal{X}}_p$ and $\tilde{\mathcal{X}}_u$. Let

$$\hat{\mathbb{E}}_p[l(g(x), 1)] = \frac{1}{\tilde{n}_p} \sum_{x \in \tilde{\mathcal{X}}_p} l(g(x), 1),$$

$$\hat{\mathbb{E}}_u[l(g(x), -1)] = \frac{1}{\tilde{n}_u} \sum_{x \in \tilde{\mathcal{X}}_u} l(g(x), -1),$$

where $\tilde{n}_p = \left\|\tilde{\mathcal{X}}_p\right\|_0$, $\tilde{n}_u = \left\|\tilde{\mathcal{X}}_u\right\|_0$. The empirical risk can be represented as:

$$\hat{R}_{PU}(g) = 2\pi_p \hat{\mathbb{E}}_p[l(g(x), 1)] + \hat{\mathbb{E}}_u[l(g(x), -1)] - \pi_p. \tag{2}$$

Hence the PU learning problem is equivalent to finding the optimal decision function

$$\hat{g}_{PU} = \arg\min_{g \in \mathcal{G}} \hat{R}_{PU}(g).$$

4.3 Re-identification Attack as PU Learning

In our re-identification attack problem, we do not have access to the positive labeled set $\tilde{\mathcal{X}}_p$. Instead we only have access to an *unlabeled* set $\mathcal{X}_u = \mathcal{X}_p \cup \mathcal{X}_n$, which includes all samples but no labels, and the sample mean of positive samples

$$\bar{x}_p = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} x.$$

As we treat the positive sample mean as a positive sample in the PU formulation, the corresponding empirical risk becomes

$$\hat{R}_{RI}(g) = 2\pi_p l(g(\bar{x}_p), 1) + \hat{\mathbb{E}}_u[l(g(x), -1)] - \pi_p.$$

We have the following theorems.

Theorem 4.1. **The noiseless case.** *Let $g(x) : \mathbb{R}^d \to \mathbb{R}$ be a linear classifier, and $l(x, y) : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ be a risk function linear to x such that $l(g(x), 1) + l(g(x), -1) = 1$. Then*

$$l(g(\bar{x}_p), 1) = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1),$$

and

$$\hat{R}_{RI}(g) = \hat{R}(g).$$

Proof. Since g is linear, and

$$\bar{x}_p = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} x,$$

we have:

$$g(\bar{x}_p) = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} g(x)$$

Since l is linear, and the labels for all positive samples and their mean are 1, we have:

$$l(g(\bar{x}_p), 1) = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1).$$

Note

$$\begin{aligned} & \hat{\mathbb{E}}_u[l(g(x), -1)] \\ = & \frac{n_p}{n_u} \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} [1 - l(g(x), 1)] + \frac{n_n}{n_u} \frac{1}{n_n} \sum_{x \in \mathcal{X}_n} l(g(x), -1) \\ = & \pi_p \left[1 - \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1) \right] + \frac{\pi_n}{n_n} \sum_{x \in \mathcal{X}_n} l(g(x), -1), \end{aligned}$$

Hence, by (1), we have

$$\begin{aligned} & \hat{R}_{RI}(g) = 2\pi_p l(g(\bar{x}_p), 1) + \hat{\mathbb{E}}_u[l(g(x), -1)] - \pi_p \\ = & \frac{\pi_p}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1) + \frac{\pi_n}{n_n} \sum_{x \in \mathcal{X}_n} l(g(x), -1) = \hat{R}(g). \end{aligned}$$

Note that in general, $\hat{R}_{PU}(g)$ is only an approximation of $\hat{R}(g)$ and the error depends on how well the average risk of the *labeled* positive samples $\tilde{\mathcal{X}}_p$ approximates that of the true positive samples. This is one of the reasons why PU algorithms tend to perform badly with very few labeled positive samples. However, Theorem 4.1 states that with a noiseless positive sample mean, $\hat{R}_{RI}(g)$ is exactly the same as $\hat{R}(g)$ with a linear classifier and a linear risk function. This partially explains why the release of a sample mean can potentially yield important information for re-identification.

Remark: Although $\hat{R}_{RI}(g) = \hat{R}(g)$, Theorem 4.1 does not guarantee that a PU algorithm will converge to the optimal classifier of learning from complete data. The PU formulation always has a potential to suffer from over-fitting due to the unlabeled samples.

As the differential privacy policy releases the sample mean with additional Laplacian noise, we also consider the noisy case.

Theorem 4.2. **The noisy case.** *Let $g(x) : \mathbb{R}^d \to \mathbb{R}$ be a linear classifier, and $l(x, y) : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ be a risk function linear to x such that $l(g(x), 1) + l(g(x), -1) = 1$. Let $W \sim Laplace(0, b)$ denote Laplacian noise with mean zero and scale factor b. Then*

$$\mathbb{E}[l(g(\bar{x}_p + W), 1)] = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1).$$

Let the corresponding empirical risk be

$$\hat{R}_{RI}(g, W) = 2\pi_p l(g(\bar{x}_p + W), 1) + \hat{\mathbb{E}}_u[l(g(x), -1)] - \pi_p,$$

then

$$\mathbb{E}[\hat{R}_{RI}(g, W)] = \hat{R}(g).$$

Proof. Since g and l are linear, we have

$$\begin{aligned} \mathbb{E}[l(g(\bar{x}_p + W), 1)] &= l(\mathbb{E}[g(\bar{x}_p + W)], 1) \\ &= l(g(\mathbb{E}[\bar{x}_p + W]), 1) \\ &= l(g(\bar{x}_p), 1) \end{aligned}$$

Further based on Theorem 4.1, we have

$$\mathbb{E}[l(g(\bar{x}_p + W), 1)] = \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1).$$

Hence, by the same argument in Theorem 4.1, we have

$$\mathbb{E}[\hat{R}_{RI}(g, W)] = \hat{R}(g).$$

Note that the expectation in Theorem 4.2 is with respect to the random Laplacian noise. For each instantiation of noise, the two empirical risks are *not* the same.

In practice, a non-linear classifier can potentially outperform a linear classifier, especially in the noisy case. We show that by establishing $\hat{R}_{RI}(g)$ as an upper bound of $\hat{R}(g)$, a classifier that minimizes the former will also reduce the risk funciton for the problem of learning from complete data.

Theorem 4.3. **Non-linear classifier.** *Let $g(x) : \mathbb{R}^d \to \mathbb{R}$ be a linear classifier, and $l(x, y) : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ be a risk function linear to x. Let $f(x) : \mathbb{R}^d \to \mathbb{R}$ be a non-linear classifier, and $k(x, y) : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ be a risk function such that $k(f(x), 1) + k(f(x), -1) = 1$. Let*

$$\hat{R}_{RI}(f, k) = 2\pi_p k(f(\bar{x}_p), 1) + \hat{\mathbb{E}}_u[k(f(x), -1)] - \pi_p.$$

If $k(f(x), 1)$ is concave with respect to x, and

$$\begin{aligned} l(g(x), 1) \le k(f(x), 1), \forall x \in \mathcal{X}_p \\ l(g(x), -1) \le k(f(x), -1), \forall x \in \mathcal{X}_n. \end{aligned} \tag{3}$$

Then

$$\hat{R}(g) \le \hat{R}_{RI}(f, k).$$

Proof. By the same argument in Theorem 4.1, we have

$$\begin{aligned} & \hat{\mathbb{E}}_u[k(f(x), -1)] \\ = & \pi_p - \frac{\pi_p}{n_p} \sum_{x \in \mathcal{X}_p} k(f(x), 1) + \frac{\pi_n}{n_n} \sum_{x \in \mathcal{X}_n} k(f(x), -1). \end{aligned}$$

Since $k(f(x), 1)$ is concave, we have

$$k(f(\bar{x}_p), 1) \ge \frac{1}{n_p} \sum_{x \in \mathcal{X}_p} k(f(x), 1),$$

so we have

$$\hat{R}_{RI}(f, k) \ge \frac{\pi_p}{n_p} \sum_{x \in \mathcal{X}_p} k(f(x), 1) + \frac{\pi_n}{n_n} \sum_{x \in \mathcal{X}_n} k(f(x), -1).$$

By (3), we have:

$$\hat{R}_{RI}(f, k) \ge \frac{\pi_p}{n_p} \sum_{x \in \mathcal{X}_p} l(g(x), 1) + \frac{\pi_n}{n_n} \sum_{x \in \mathcal{X}_n} l(g(x), -1)] = \hat{R}(g).$$

We demonstrate the feasibility of re-identification attack with a positive sample mean using the nnPU algorithm [14] for PU Learning. Our result shows that with a 3-layer-perceptron classifier, it achieves **85.7% re-identification accuracy** on the MNIST dataset. This is surprising as nnPU is known to perform badly for the PU Learning problem when the number of labeled positive samples, $\tilde{n}_p$, is small. Table 1 shows nnPU's performance for the PU Learning problem with the same 3-layer-perceptron classifier on the MNIST dataset. The classification accuracy of nnPU is almost linear with respect to $\log \tilde{n}_p$. When $\tilde{n}_p < 300$, the accuracy drops drastically from 93.3% to 51.9%.

$\tilde{n}_p$	1000	300	100	30	10	3	1
accuracy	94.4	93.3	87.2	79.6	64.3	59.1	51.9

Table 1: nnPU's performance for the PU Learning problem on the MNIST dataset. The classification accuracy is low with only a few labeled positive samples.

5 THE GROWPU ALGORITHM

We propose a novel algorithm to exploit the unique property of sample mean and further increase the accuracy of the re-identification attack. Exhibit 1 describes the algorithm. It consists of three stages: pre-train, growth and fine-tune. Table 2 shows the hyper-parameters of the algorithm.

b:	training batch size
π_{pre}:	target proportion of classified positive data for pre-train
π_{grow}:	target proportion of classified positive data for growth
T_{pre}:	number of maximum iterations for pre-train
T_{tune}:	number of maximum iterations for fine-tune
α:	weight discount buffer size
γ:	sample weight discount factor

Table 2: Hyper-parameters of the growPU algorithm.

A crucial problem in minimizing the risk with one-sided labels is overfitting, where all unlabeled samples are classified as *negative*. It is known that the PU learning algorithm, uPU [7] suffers from overfitting. The nnPU algorithm [14] deals with overfitting by forcing the proportion of classified positive samples to stay close to the given π_p. The key idea of growPU, however, is to *exploit* the overfitting process at the pre-train stage, which generates high-quality pseudo-labels for a very small set of samples. The high quality of the initial set seems to be a special property of the sample mean, as it does not hold for a random positive sample.

The algorithm then *grows* from this initial set by modifying the risk function to include the pseudo-labels:

$$\hat{R}_{RI}(g) = \frac{\pi_p}{\hat{n}_p + 1}\Big[l(g(\bar{x}_p), 1) + \sum_{x \in \hat{\mathcal{X}}_p} l(g(x), 1)\Big] + \frac{\pi_n}{\hat{n}_n} \sum_{x \in \hat{\mathcal{X}}_n} l(g(x), -1),$$

where the set $\hat{\mathcal{X}}_p$ contains the positive pseudo-labels and the set $\hat{\mathcal{X}}_n$ contains the negative pseudo-labels as defined in Exhibit 3. The weight w_n defined in Exhibit 3 is applied to balance the training of the two sets. The pseudo-labels seem to contribute new representative features not obtainable from the sample mean alone, hence increasing the accuracy.

The fine-tune stage is similar to nnPU where the proportion of samples classified as positive stays close to π_p. Figure 4(b) shows that the uPU algorithm suffers from severe overfitting as its proportion of classified positive samples approaches zero. The nnPU alogrithm, on the other hand, has its proportion of classified positive samples oscillate around $\pi_p = 0.5$. Figure 4(a) shows that growPU algorithm, in contrast, overfits in the first 2000 rounds, then grows and oscillates around $\pi_p = 0.5$.

Exhibit 1: The growPU Algorithm

Input
  π_p: the proportion of positive sample
  $\bar{x}_p$: positive sample mean
  $\mathcal{X}_u$: unlabeled samples

Initialize
  C: the classifier

Pre-train
  for iteration number $t = 1$ to T_{pre}
    $\mathcal{X}, \mathcal{Y} \leftarrow PretrainSampleSection(\bar{x}_p, \mathcal{X}_u, b)$
    $Train(model = C, data = \mathcal{X}, label = \mathcal{Y})$
    $\hat{\mathcal{X}}_p \leftarrow \{x : C(x) = 1, x \in \mathcal{X}\}$
    if $\|\hat{\mathcal{X}}_p\|_0 / \|\mathcal{X}\|_0 \le \pi_{pre}$:
      $break$

Growth
  $isFineTune = False$
  while 1:
    $\mathcal{X}, \mathcal{Y}, \mathcal{W} \leftarrow GrowSampleSection(\bar{x}_p, \mathcal{X}_u, b, isFineTune)$
    $Train(model = C, data = \mathcal{X}, label = \mathcal{Y}, weight = \mathcal{W})$
    $\hat{\mathcal{X}}_p \leftarrow \{x : C(x) = 1, x \in \mathcal{X}\}$
    if $\|\hat{\mathcal{X}}_p\|_0 / \|\mathcal{X}\|_0 \ge \pi_{grow}$:
      $break$

Fine-tune
  $isFineTune = True$
  for iteration number $t = 1$ to T_{tune}
    $\mathcal{X}, \mathcal{Y}, \mathcal{W} \leftarrow GrowSampleSection(\bar{x}_p, \mathcal{X}, b, isFineTune)$
    $Train(model = C, data = \mathcal{X}, label = \mathcal{Y}, weight = \mathcal{W})$

Exhibit 2: PretrainSampleSelection

Select Samples
  $\tilde{\mathcal{X}} \leftarrow b$ random samples from $\mathcal{X}_u$
  $\mathcal{X} \leftarrow \tilde{\mathcal{X}} + [\bar{x}_p, ...]$ of length b
  $\mathcal{Y} \leftarrow [-1, ...]$ of length b + $[1, ...]$ of length b
  Return $\mathcal{X}, \mathcal{Y}$

Figure 4 shows the proportion of classified-positive samples and re-identification accuracy of the three algorithms respectively. We observe that the accuracy of the uPU algorithm is severely hurt by overfitting while the growPU algorithm *benefits* from the initial overfitting to achieve a higher accuracy than nnPU.

6 EXPERIMENT

We demonstrate the re-identification accuracy of nnPU, uPU and the proposed growPU algorithm on the MNIST dataset [5] of handwritten numbers and the online user behavioral dataset analyzed in Section 3. We use the most challenging pair, '5' vs. '3', from the MNIST dataset, and randomly select 5,000 samples for each class. The online user

Exhibit 3: GrowSampleSelection

Select Samples

$\tilde{\mathcal{X}} \leftarrow b$ random samples from $\mathcal{X}_u$

$\hat{\mathcal{X}}_p \leftarrow \{x : x \in \tilde{\mathcal{X}}, C(x) = 1\}$

$\hat{\mathcal{X}}_n \leftarrow \{x : x \in \tilde{\mathcal{X}}, C(x) = -1\}$

$\hat{n}_p \leftarrow \left\|\hat{\mathcal{X}}_p\right\|_0$; $\hat{n}_n \leftarrow \left\|\hat{\mathcal{X}}_n\right\|_0$

$\hat{\mathcal{Y}}_p \leftarrow [1, ...]$ of length $\hat{n}_p$; $\hat{\mathcal{Y}}_n \leftarrow [-1, ...]$ of length $\hat{n}_n$

$w_p = 1$; $w_n = \frac{(1-\pi_p)(\hat{n}_p+1)}{\pi_p \hat{n}_n}$

if isFineTune :

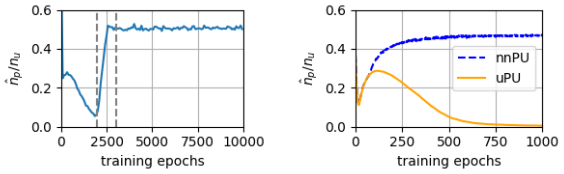
if $\frac{\hat{n}_p}{\hat{n}_p+\hat{n}_n} \leq \pi_p - \alpha : w_n = w_n * \gamma$

if $\frac{\hat{n}_p}{\hat{n}_p+\hat{n}_n} \geq \pi_p + \alpha : w_p = w_p * \gamma$

$\mathcal{X} \leftarrow \hat{\mathcal{X}}_n + \hat{\mathcal{X}}_p + [\bar{x}_p]$; $\mathcal{Y} \leftarrow \hat{\mathcal{Y}}_n + \hat{\mathcal{Y}}_p + [1]$

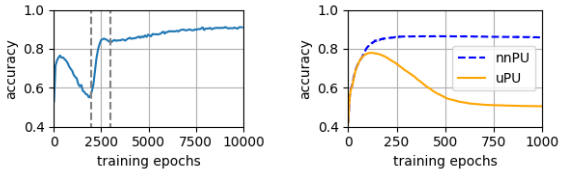
$\mathcal{W} \leftarrow [w_n, ...]$ of length $\hat{n}_n + [w_p, ...]$ of length $\hat{n}_p + 1$

Return $\mathcal{X}, \mathcal{Y}, \mathcal{W}$



(a) Proportion of classified-positive samples of growPU. It first overfits, then grows the proportion of classified-positive to π_p.

(b) Proportion of classified-positive samples of nnPU and uPU. nnPU maintains the set around π_p, while uPU overfits.



(c) Re-identification accuracy of growPU. It benefits from overfitting and continues to improve at the fine-tune stage.

(d) Re-identification accuracy of nnPU and uPU. nnPU is stable while uPU suffers from overfitting.

Figure 4: Evolution of proportion of classified-positive samples and re-identification accuracy of growPU, nnPU and uPU.

behavioral dataset contains 10,000 samples from each class. Both datasets are balanced, i.e. $\pi_p = \pi_n = 0.5$.

6.1 Experiment Setup

6.1.1 Hyper-parameters. Table 3 shows the choice of hyper-parameters for the growPU algorithm. The parameter π_{pre} controls the termination of the pre-train stage, and is set to 5% for all experiments. At the fine-tune stage, the parameter α introduces a tolerance of oscillation around the boundary, and the parameter γ is used to balance the two classes. We set $\gamma = 0.5$ and $\alpha = 5\%$ for all experiments.

The parameter T_{pre} controls the maximum iterations at the pre-train stage, and T_{tune} controls that at the fine-tune stage. For nnPU and uPU, the classifiers are trained for 1,000 epochs each. We use the same batch size, b, for all three algorithms.

Parameter	MNIST	online behavioral
b	500	1000
π_{pre}	0.05	0.05
π_{grow}	0.45	0.45
T_{pre}	10000	10000
T_{tune}	10000	10000
α	0.05	0.05
γ	0.5	0.5

Table 3: GrowPU hyper-parameter selection

6.1.2 Variations of Sample Mean.

(1) Noiseless sample-mean: The baseline case where no noise is added in computing the sample-mean of all or a subset of the positive samples.

(2) Distribution mean: Instead of the empirical average of the positive samples, a GAN is trained on the positive samples and a mean is simulated. This treats the case where the sample mean is released, but of a different set of samples from the same distribution.

(3) Noisy sample-mean: Laplacian noise $Laplace(0, \frac{\max(x)-\min(x)}{n_p \cdot \epsilon})$ is added to $\bar{x}_p$ to guarantee ϵ-differential privacy, where $\frac{\max(x)-\min(x)}{n_p}$ is the sensitivity of the mean function. For instance, with the MNIST dataset, $\max(x) - \min(x) \leq 1 - (-1) = 2$. With $n_p = 5000$ and $\epsilon = 0.01$, which is a strong differential privacy criterion, we add noise $Laplace(0, \frac{2}{5000 \cdot 0.01})$.

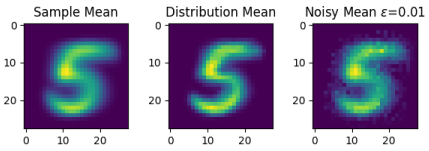


Figure 5: Variations of sample mean for the MNIST data.

6.1.3 The Classifiers.

(1) Multi-layer-perceptron (MLP) classifier: A MLP classifier has an input layer, one or more hidden layers and an output layer. Drop-out rate of 20% is added to each hidden layer.

(2) Linear classifier: A special case of 2LP classifier. It consists of one input layer, one output layer and no hidden layer. The output layer conducts linear transformation of inputs, following by a sigmoid function to output.

Table 4 shows the re-identification accuracy of growPU and nnPU using different underlying MLP classifiers with noiseless sample-mean. We choose the best-performing MLP classifiers for each algorithm and dataset in the following experiments. As uPU's performance has large variation due to overfitting, we select for uPU the classifier that performs the best with nnPU.

Re-identification accuracy (%)				
Classifier	nnPU-MNIST	nnPU-online behavioral	growPU-MNIST	growPU-online behavioral
3lp-100	83.8	77.2	90.3	84.6
3lp-200	84.6	**77.5**	91.3	**85.4**
3lp-300	**85.7**	77.4	**92.0**	85.1
4lp-100	81.7	70.5	73.6	84.4
4lp-200	79.4	72.6	78.1	84.5
4lp-300	77.9	70.3	84.9	85.4
5lp-100	78.0	59.4	79.1	84.9
5lp-200	78.4	55.1	84.8	85.4
5lp-300	78.7	54.1	90.2	85.3

Table 4: Re-identification accuracy of growPU and nnPU with different MLP classifiers and noiseless sample-mean. The best performance is highlighted in bold.

6.2 Comparison on the MNIST dataset

Figure 6 visualizes the first two PCA dimensions of all '5' and '3' samples. Table 5 shows the performance of each 'algorithm | classifier' combination. For instance, 'growPU | 3lp' uses the growPU algorithm and a a 3LP classifier with hidden layer size 300, as selected from Table 4. We also report the results of K-means clustering and an 'oracle' algorithm that runs supervised learning on the data, which acts as an upper bound on the accuracy.

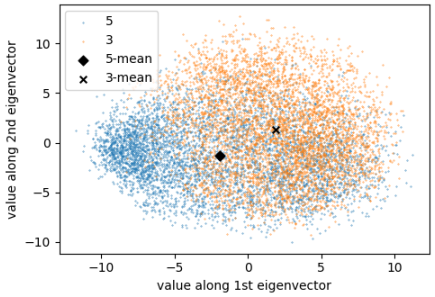


Figure 6: Values of the first two PCA dimensions of '5' and '3' are largely overlapping.

We observe that growPU with a linear classifier achieves the best performance for the noiseless sample mean, and growPU with a 3LP classifier achieves the best performance for the distribution and noisy sample means. Apart from the distribution case, both re-identification accuracies exceed 90%. Since K-means clustering has an accuracy of 68% and the oracle algorithm has an accuracy of 98.4%, the release of a sample-mean, even a noisy one, increases the re-identification accuracy much closer to that of the oracle, who has complete information.

The nnPU algorithm with 3LP performs better than that with a linear classifier. The highest accuracy of nnPU is $5-8\%$ below that of growPU. The accuracy of uPU with a linear classifier is another $6-11\%$ below the highest of nnPU, and uPU with 3LP degrades to a random guess as the large number of parameters in a 3-layer network encourages overfitting.

Re-identification accuracy (%)			
Algorithm	Noiseless	Distribution	Noisy ϵ=0.01
growPU | Linear	**93.6**	84.6	91.8
nnPU | Linear	79.9	79.2	81.0
uPU | Linear	79.4	70.3	75.8
oracle | Linear	95.0		
growPU | 3lp	92.0	**86.3**	**91.9**
nnPU | 3lp	85.7	81.3	85.8
uPU | 3lp	50.5	50.2	50.1
oracle | 3lp	98.4		
K-means	68.0		

Table 5: Re-identification accuracy on the '5' vs '3' MNIST dataset. The highest accuracy for each category is highlighted in bold.

In addition, we run experiments for the scenario where only a subset of the unlabeled samples are available at the time of training. For the MNIST dataset, we train the algorithms on $1,000$ unlabeled samples, and attempt re-identification of all $10,000$ samples. Table 6 shows the re-identification accuracy on the $10,000$ samples. We observe that most entries for growPU show a slight degradation between $0.3-2\%$, but growPU remains the one with the highest accuracy.

Re-identification accuracy (%)			
Algorithm	Noiseless	Distribution	Noisy ϵ=0.01
growPU | Linear	**93.3**	84.1	**90.9**
nnPU | Linear	79.0	81.6	79.8
uPU | Linear	72.4	67.2	70.4
growPU | 3lp	90.1	**85.7**	90.2
nnPU | 3lp	84.2	81.6	82.8
uPU | 3lp	51.2	51.9	50.5

Table 6: Re-identification accuracy on the '5' vs '3' MNIST dataset with only a subset of unlabeled samples. The highest accuracy for each category is highlighted in bold.

6.3 Comparison on the Online Behavioral Dataset

Table 7 shows the performance of 'algorithm | classifier' combinations on the online user behavioral dataset. In this table, 'growPU | 3lp' uses the growPU algorithm and a 3LP classifier with hidden layer size 200, as selected for this dataset in Table 4.

We observe that the re-identification accuracy for this dataset is overall lower than that for the MNIST dataset. This is potentially due to the interspersed clusters and extreme sparsity characteristic of this dataset, as illustrated in Section 3.4. We note that the oracle only achieves 89.2% accuracy with a 3LP classifier, and K-means clustering only achieves 60% accuracy.

The growPU algorithm with a linear classifier achieves the highest accuracy over all variations of sample mean, a mere 1% below the oracle's performance with a noiseless sample mean, and 4.5% with noise. With a 3LP classifier, the accuracy is comparable, but $1-3\%$ lower over all variations of sample mean.

The nnPU algorithm still performs better with a 3LP classifier. However, the highest accuracy achieved by nnPU is $5-11\%$ below

that of growPU for both the noiseless and noisy case. The accuracy of uPU with a linear classifier is another 11 – 13% below the highest of nnPU. This online behavioral dataset, with its more challenging features, seems to have enlarged the difference in performance of the three algorithms.

Re-identification accuracy (%)			
Algorithm	Noiseless	Distribution	Noisy ϵ=0.01
growPU | Linear	**88.1**	**81.0**	**84.6**
nnPU | Linear	73.0	68.7	73.4
uPU | Linear	64.9	62.9	65.2
oracle | Linear	89.1		
growPU | 3lp	85.4	80.1	82.6
nnPU | 3lp	77.5	76.0	76.2
uPU | 3lp	50.7	50.6	50.1
oracle | 3lp	89.2		
K-means	60.0		

Table 7: Re-identification accuracy on the online behavioral dataset. The highest accuracy for each category is highlighted in bold.

For the scenario where only a subset of the unlabeled samples are available, we train the algorithms on 10% of unlabeled samples and attempt re-identification of all samples. Table 8 shows the re-identification accuracy on all 20, 000 samples. We observe that the performance is very similar, but growPU remains the one with the highest accuracy.

Re-identification accuracy (%)			
Algorithm	Noiseless	Distribution	Noisy ϵ=0.01
growPU | Linear	**88.0**	**83.5**	**83.5**
nnPU | Linear	72.7	71.0	72.9
uPU | Linear	71.1	62.9	72.6
growPU | 3lp	83.8	80.9	81.4
nnPU | 3lp	77.8	76.3	76.0
uPU | 3lp	51.8	50.6	50.7

Table 8: Re-identification accuracy on the online behavioral dataset with only a subset of unlabeled samples. The highest accuracy for each category is highlighted in bold.

7 CONCLUSION

In this paper, we demonstrate that with a one-sided noisy sample mean, a highly effective re-identification attack can be devised to recover the confidential class labels. We found that the re-identification attack problem can be formulated as a generalized positive-unlabeled learning (PU learning) problem, and is closely related to the problem of learning with complete data. Experiments on benchmark datasets and an online user behavioral dataset show that the proposed algorithm consistently achieves high re-identification accuracy.

REFERENCES

[1] Martin Abadi, 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*. 308–318.

[2] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 132–149.

[3] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. 2008. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)* 26, 2 (2008), 1–26.

[4] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks* 20, 3 (2009), 542–542.

[5] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.

[6] Marthinus Du Plessis, Gang Niu, and Masashi Sugiyama. 2015. Convex formulation for learning from positive and unlabeled data. In *International Conference on Machine Learning*. 1386–1394.

[7] Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of learning from positive and unlabeled data. In *Advances in neural information processing systems*. 703–711.

[8] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*. Springer, 1–19.

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

[10] Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 213–220.

[11] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. Unsupervised learning. In *The elements of statistical learning*. Springer, 485–585.

[12] Jim Isaak and Mina J Hanna. 2018. User data privacy: Facebook, Cambridge Analytica, and privacy protection. *Computer* 51, 8 (2018), 56–59.

[13] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*. 3581–3589.

[14] Ryuichi Kiryo, Gang Niu, Marthinus C du Plessis, and Masashi Sugiyama. 2017. Positive-unlabeled learning with non-negative risk estimator. In *Advances in neural information processing systems*. 1675–1685.

[15] Samuli Laine and Timo Aila. 2016. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242* (2016).

[16] Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, Vol. 3. 2.

[17] Naresh K Malhotra, Sung S Kim, and James Agarwal. 2004. Internet users' information privacy concerns (IUIPC): The construct, the scale, and a causal model. *Information systems research* 15, 4 (2004), 336–355.

[18] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence* 41, 8 (2018), 1979–1993.

[19] Noman Mohammed, Rui Chen, Benjamin Fung, and Philip S Yu. 2011. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 493–501.

[20] Rathindra Sarathy and Krishnamurty Muralidhar. 2011. Evaluating Laplace noise addition to satisfy differential privacy for numeric data. *Trans. Data Privacy* 4, 1 (2011), 1–17.

[21] Christof Strauch, Ultra-Large Scale Sites, and Walter Kriha. 2011. NoSQL databases. *Lecture Notes, Stuttgart Media University* 20 (2011), 24.

[22] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*. 478–487.

[23] Lei Xu, Chunxiao Jiang, Jian Wang, Jian Yuan, and Yong Ren. 2014. Information security in big data: privacy and data mining. *Ieee Access* 2 (2014), 1149–1176.

[24] Rui Xu and Donald Wunsch. 2005. Survey of clustering algorithms. *IEEE Transactions on neural networks* 16, 3 (2005), 645–678.

[25] Rui Xu and Don Wunsch. 2008. *Clustering*. Vol. 10. John Wiley & Sons.

[26] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 3861–3870.

[27] Zhen-Yu Zhang, Peng Zhao, Yuan Jiang, and Zhi-Hua Zhou. 2019. Learning from incomplete and inaccurate supervision. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1017–1025.

[28] Xiaojin Zhu and Andrew B Goldberg. 2009. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning* 3, 1 (2009), 1–130.