Learning Privately from Multiparty Data

Jihun Hamm HAMMJ@CSE.OHIO-STATE.EDU
Dept. Computer Science and Engineering, Ohio State University, Columbus, OH 43210, USA

Paul Cao YIC242@ENG.UCSD.EDU
Dept. Computer Science and Engineering, UC-San Diego, La Jolla, CA 92093, USA

Mikhail Belkin MBELKIN@CSE.OHIO-STATE.EDU
Dept. Computer Science and Engineering, Ohio State University, Columbus, OH 43210, USA

Abstract

Learning a classifier from private data collected by multiple parties is an important problem that has many potential applications. How can we build an accurate and differentially private global classifier by combining locally-trained classifiers from different parties, without access to any party's private data? We propose to transfer the 'knowledge' of the local classifier ensemble by first creating labeled data from auxiliary unlabeled data, and then train a global ϵ-differentially private classifier. We show that majority voting is too sensitive and therefore propose a new risk weighted by class probabilities estimated from the ensemble. Relative to a non-private solution, our private solution has a generalization error bounded by $O(\epsilon^{-2}M^{-2})$ where M is the number of parties. This allows strong privacy without performance loss when M is large, such as in crowdsensing applications. We demonstrate the performance of our method with realistic tasks of activity recognition, network intrusion detection, and malicious URL detection.

1. Introduction

Consider the problem of performing machine learning with data collected by multiple parties. In many settings, the parties may not wish to disclose the private information. For example, the parties can be medical institutions who aim to perform collaborative research using sensitive patient information they hold. For another example, the parties can be computer users who aim to collectively build a malware detector without sharing their usage data. A conventional approach to learning from multiparty data is to first collect data from all parties and then process them centrally. When privacy is a major concern, this approach is not always an appropriate solution since it is vulnerable to attacks during transmission, storage, and processing of data. Instead, we will consider a setting in which each party trains a *local* classifier from its private data without sending the data. The goal is to build a *global* classifier by combining local classifiers efficiently and privately. We expect the global classifier to be more accurate than individual local classifiers, as it has access to more information than individual classifiers.

This problem of aggregating classifiers was considered in (Pathak et al., 2010), where the authors proposed averaging of the *parameters* of local classifiers to get a global classifier. To prevent the leak of private information from the averaged parameters, the authors used a differentially private mechanism. Differential privacy measures maximal change in the probability of any outcome of a procedure when any item is added to or removed from a database. It provides a strict upper bound on the privacy loss against any adversary (Dwork & Nissim, 2004; Dwork et al., 2006; Dwork, 2006). Parameter averaging is a simple and practical procedure that can be implemented by Secure Multiparty Computation (Yao, 1982). However, averaging is not applicable to classifiers with non-numerical parameters such as decision trees, nor to a collection of different classifier types. This raises the question if there are more flexible and perhaps better ways of aggregating local classifiers privately.

In this paper, we propose a method of building a global differentially private classifier from an ensemble of local classifier in two steps (see Fig. 1.) In the first step, locally-trained classifiers are collected by a trusted entity. A naive approach to use the collected classifiers is to release (the parameters of) the classifiers after sanitization by differentially private mechanisms, which is impractical

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

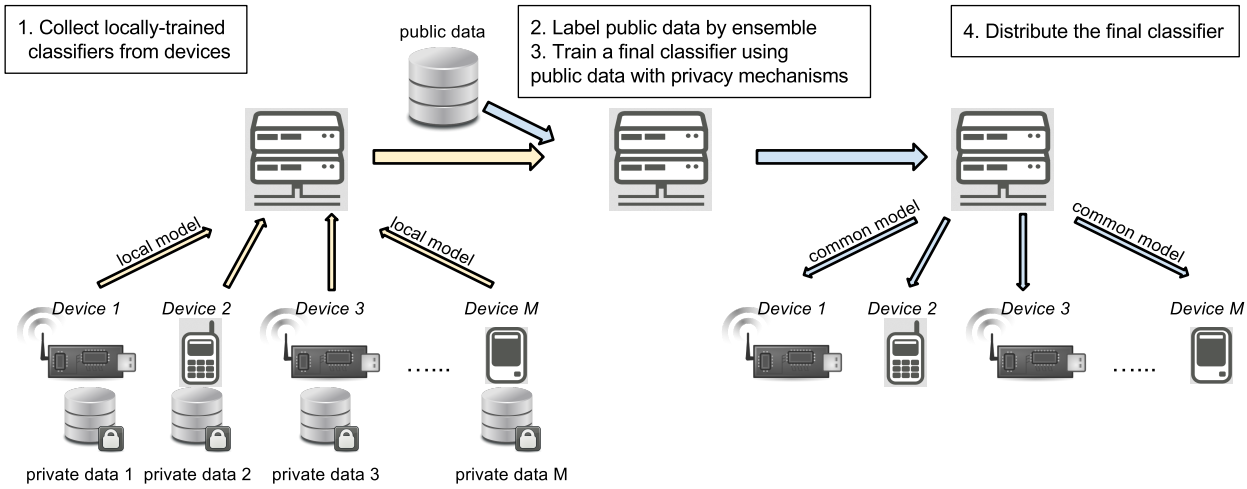

Figure 1. Workflow of the proposed algorithm. Assume that the parties are smart devices. Each party holds a small amount of private data and uses the data to train a local classifier. The ensemble of local classifiers collected then generates labels for auxiliary data, which in turn are used for training a global classifier. The final classifier is released after sanitization for privacy.

(Sec. 3.2.) Instead, we use the classifier ensemble to generates (pseudo)labels for auxiliary *unlabeled* data, thus transferring the knowledge of the ensemble to the auxiliary data. In the second step, we use the labeled auxiliary data to find an empirical risk minimizer, and release a differentially private classifier using output perturbation (Chaudhuri et al., 2011).

When generating labels for auxiliary data using an ensemble of classifiers, majority voting is the simplest choice. However, we show quantitatively that a global classifier trained from majority-voted labels is highly sensitive to individual votes of local classifiers. Consequently, the final classifier after differentially-private sanitization suffer a heavy loss in its performance. To address this, we propose a new risk insensitive to individual votes, where each sample is *weighted* by the confidence of the ensemble. We provide an interpretation of the weighted risk in terms of random hypothesis of an ensemble (Breiman, 1996a) in contrast to deterministic labeling rule of majority voting. One of our main results is in Theorem 4: we can achieve ϵ-differential privacy with a generalization error of $O(\epsilon^{-2}M^{-2})$ and $O(N^{-1})$ terms, relative to the expected risk of a non-private solution, where M is the number of parties and N is the number of samples in auxiliary data. This result is especially useful in a scenario where there are a large number of parties with weak local classifiers such as a group of connected smart devices with limited computing capability. We demonstrate the performance of our approach with several realistic tasks: activity recognition, network intrusion detection, and malicious URL detection. The results show that it is feasible to achieve both accuracy and privacy with a large number of parties.

To summarize, we propose a method of building a global differentially private classifier from locally-trained classifiers of multiple parties without access to their private data. The proposed method has the following advantages: 1) it can use local classifiers of any (mixed) types and therefore is flexible; 2) its generalization error converges to that of a non-private solution with a fast rate of $O(\epsilon^{-2}M^{-2})$ and $O(N^{-1})$; 3) it also provides ϵ-differential privacy to all samples of a party and not just a single sample.

In Sec. 2, we formally describe privacy definitions. In Sec. 3, we discuss the first step of leveraging unlabeled data, and in Sec. 4, we present the second step of finding a global private classifier via empirical risk minimization in two different forms. In Sec. 5, we discuss related works. We evaluate the methods with real tasks in Sec. 6 and conclude the paper in Sec. 7. The supplementary material contains omitted proofs and several extensions of the algorithms.

2. Preliminary

2.1. Differential privacy

A randomized algorithm that takes data $\mathcal{D}$ as input and outputs a function f is called ϵ-differentially private if

$$\frac{P(f(\mathcal{D}) \in \mathcal{S})}{P(f(\mathcal{D}') \in \mathcal{S})} \leq e^{\epsilon} \tag{1}$$

for all measurable $\mathcal{S} \subset \mathcal{T}$ of the output range and for all datasets $\mathcal{D}$ and $\mathcal{D}'$ differing in a single item, denoted by $\mathcal{D} \sim \mathcal{D}'$. That is, even if an adversary knows the whole dataset $\mathcal{D}$ except for a single item, she cannot infer much about the unknown item from the output f of the algorithm.

When an algorithm outputs a real-valued vector $f \in \mathbb{R}^D$, its global L_2 sensitivity (Dwork et al., 2006) can be defined as

$$S(f) = \max_{\mathcal{D} \sim \mathcal{D}'} \|f(\mathcal{D}) - f(\mathcal{D}')\| \quad (2)$$

where $\|\cdot\|$ is the L_2 norm. An important result from (Dwork et al., 2006) is that a vector-valued output f with sensitivity $S(f)$ can be made ϵ-differentially private by perturbing f with an additive noise vector η whose density is

$$P(\eta) \propto e^{-\frac{\epsilon}{S(f)}\|\eta\|}. \quad (3)$$

2.2. Output perturbation

When a classifier which minimizes empirical risk is released in public, it leaks information about the training data. Such a classifier can be sanitized by perturbation with additive noise calibrated to the sensitivity of the classifier, known as output perturbation method (Chaudhuri et al., 2011). Specifically, the authors show the following. If w_s is the minimizer of the regularized empirical risk

$$R_S^\lambda(w) = \frac{1}{N} \sum_{(x,y)\in S} l(h(x;w),y) + \frac{\lambda}{2}\|w\|^2, \quad (4)$$

Then the perturbed output $w_p = w_s + \eta,\ p(\eta) \propto e^{-\frac{N\lambda\epsilon}{2}\|\eta\|}$ is ϵ-differentially private for a single sample. Output perturbation was used to sanitize the averaged parameters in (Pathak et al., 2010). We also use output perturbation to sanitize global classifiers. One important difference of our setting to previous work is that we consider ϵ-differential privacy of all samples of a party, which is much stronger than ϵ-differential privacy of a single sample.

There are conditions on the loss function for this guarantee to hold. We will assume the following conditions for our global classifier[1] similar to (Chaudhuri et al., 2011).

- The loss-hypothesis has a form $l(h(x;w),v)) = l(vw^T\phi(x))$, where $\phi : \mathcal{X} \to \mathbb{R}^d$ is a fixed map. We will consider only linear classifiers $l(vw^Tx)$, where any nonlinear map ϕ is absorbed into the d-dimensional features.
- The surrogate loss $l(\cdot)$ is convex and continuously differentiable.
- The derivative $l'(\cdot)$ is bounded: $|l'(t)| \le 1,\ \forall t \in \mathbb{R}$, and c-Lipschitz: $|l'(s) - l'(t)| \le c|s-t|,\ \forall s,t \in \mathbb{R}$.
- The features are bounded: $\sup_{x\in\mathcal{X}} \|x\| \le 1$.

These conditions are satisfied by, e.g., logistic regression loss ($c = 1/4$) and approximate hinge loss.

[1] Local classifiers are allowed to be of any type.

3. Transferring knowledge of ensemble

3.1. Local classifiers

In this paper, we treat local classifiers as M black boxes $h_1(x), ..., h_M(x)$. We assume that a local classifier $h_i(x)$ is trained using its private i.i.d. training set $S^{(i)}$

$$S^{(i)} = \{(x_1^{(i)}, y_1^{(i)}),\ \dots,\ (x_{N_i}^{(i)}, y_{N_i}^{(i)})\}, \quad (5)$$

where $(x_j^{(i)}, y_j^{(i)}) \in \mathcal{X} \times \{-1, 1\}$ is a sample from a distribution $P(x, y)$ common to all parties. We consider binary labels $y \in \{-1, 1\}$ in the main paper, and present a multiclass extension in Appendix B of the supplementary material.

This splitting of training data across parties is similar to the Bagging procedure (Breiman, 1996a) with some differences. In Bagging, the training set $S^{(i)}$ for party i is sampled *with replacement* from the whole dataset $\mathcal{D}$, whereas in our setting, the training set is sampled *without replacement* from $\mathcal{D}$, more similar to the Subagging (Politis et al., 1999) procedure.

3.2. Privacy issues of direct release

In the first step of our method, local classifiers from multiple parties are first collected by a trusted entity. A naive approach to use the ensemble is to directly release the local classifier parameters to the parties after appropriate sanitization. However, this is problematic in efficiency and privacy. Releasing all M classifier parameters is an operation with a constant sensitivity, as opposed to releasing insensitive statistics such as an average whose sensitivity is $O(M^{-1})$. Releasing the classifiers requires much stronger perturbation than necessary, incurring steep loss of performance of sanitized classifiers. Besides, efficient differentially private mechanisms are known only for certain types of classifiers so far (see (Ji et al., 2014) for a review.) Another approach is to use the ensemble as a service to make predictions for test data followed by appropriate sanitization. Suppose we use majority voting to provide a prediction for a test sample. A differentially private mechanism such as Report Noisy Max (Dwork & Roth, 2013) can be used to sanitize the votes for a *single* query. However, answering several queries requires perturbing all answers with noise linearly proportional to the number of queries, which is impractical in most realistic settings.

3.3. Leveraging auxiliary data

To address the problems above, we propose to transfer the knowledge of the ensemble to a global classifier using auxiliary unlabeled data. More precisely, we use the ensemble to generate (pseudo)labels for the auxiliary data, which in turn are used to train a global classifier. Compared to directly releasing local classifiers, releasing a global classifier

trained on auxiliary data is a much less sensitive operation with $O(M^{-1})$ (Sec. 4.4) analogous to releasing an average statistic. The number of auxiliary samples does not affect privacy, and in fact the larger the data the closer the global classifier is to the original ensemble with $O(N^{-1})$ bound (Sec. 4.4). Also, compared to using the ensemble to answer prediction queries, the sanitized global classifier can be used as many times as needed without its privacy being affected.

We argue that the availability of auxiliary unlabeled data is not an issue, since in many settings they are practically much easier to collect than labeled data. Furthermore, if the auxiliary data are obtained from public repositories, privacy of such data is not an immediate concern. We mainly focus on the privacy of local data, and discuss extensions for preserving the privacy of auxiliary data in Sec. 4.5.

4. Finding a global private classifier

We present details of training a global private classifier. As the first attempt, we use majority voting of an ensemble to assign labels to auxiliary data, and find a global classifier from the usual ERM procedure. In the second attempt, we present a better approach where we use the ensemble to estimate the posterior $P(y|x)$ of the auxiliary samples and solve a 'soft-labeled' weighted empirical minimization.

4.1. First attempt: ERM with majority voting

As the first attempt, we use majority voting of M local classifiers to generate labels of auxiliary data, and analyze its implications. Majority voting for binary classification is the following rule

$$v(x) = \begin{cases} 1, & \text{if } \sum_{i=1}^{M} I[h_i(x) = 1] \geq \frac{M}{2} \\ -1, & \text{otherwise} \end{cases}. \quad (6)$$

Ties can be ignored by assuming an odd number M of parties. Regardless of local classifier types or how they are trained, we can consider the majority vote of the ensemble $\{h_1, ..., h_M\}$ as a *deterministic* target concept to train a global classifier.

The majority-voted auxiliary data are

$$S = \{(x_1, v(x_1)),\ ...,\ (x_N, v(x_N))\}, \quad (7)$$

where $x_i \in \mathcal{X}$ is an i.i.d. sample from the same distribution $P(x)$ as the private data. We train a global classifier by minimizing the (regularized) empirical risk associated with a loss and a hypothesis class:

$$R_S^{\lambda}(w) = \frac{1}{N} \sum_{(x,v)\in S} l(h(x; w), v) + \frac{\lambda}{2}\|w\|^2. \quad (8)$$

We use $R^{\lambda}(w)$ and $R(w)$ without the subscript S to denote expected risks with and without regularization. Algorithm 1 summarizes the procedure.

Algorithm 1 DP Ensemble by Majority-voted ERM

Input: $h_1, ..., h_M$ (local classifiers), X (auxiliary unlabeled samples), ϵ, λ
Output: w_p
Begin
  for $i = 1,\ ...\ , N$ **do**
    Generate majority voted labels $v(x_i)$ by (6)
  end for
  Find the minimizer w_s of (8) with $S = \{(x_i, v(x_i))\}$
  Sample a random vector η from $p(\eta) \propto e^{-0.5\lambda\epsilon\|\eta\|}$
  Output $w_p = w_s + \eta$

Applying output perturbation to our multiparty setting gives us the following result.

Theorem 1. *The perturbed output $w_p = w_s + \eta$ from Algorithm 1 with $p(\eta) \propto e^{-\frac{\lambda\epsilon}{2}\|\eta\|}$ is ϵ-differentially private.*

The proof of the theorem and others are in the Appendix A of the supplementary material.

4.2. Performance issues of majority voting

We briefly discuss the generalization error of majority-voted ERM. In (Chaudhuri et al., 2011), it is shown that the expected risk of an output-perturbed ERM solution w_p with respect to the risk of any reference hypothesis w_0 is bounded by two terms – one due to noise and another due to the gap between expected and empirical regularized risks. This result is applicable to the majority-voted ERM with minor modifications. The sensitivity of majority-voted ERM from Theorem 1 is $\frac{2}{\lambda}$ compared to $\frac{2}{N\lambda}$ of a standard ERM, and corresponding the error bound is

$$R(w_p) \leq R(w_0) + O(\epsilon^{-2}) + O(N^{-1}), \quad (9)$$

with high probability, ignoring other variables. Unfortunately, the bound does not guarantee a successful learning due to the constant gap $O(\epsilon^{-2})$, which can be large for a small ϵ.

What causes this is the worst-case scenario of multiparty voting. Suppose the votes of $M-1$ local classifiers are exactly ties for all auxiliary samples $\{x_1, ..., x_N\}$. If we replace a local classifier $h_i(x)$ with the 'opposite' classifier $h_i'(x) = -h_i(x)$, then the majority-voted labels $\{v_1, ..., v_N\}$ become $\{-v_1, ..., -v_N\}$, and the resultant global classifier is entirely different. However unlikely this scenario may be in reality, differential privacy requires that we calibrate our noise to the worst case sensitivity.

4.3. Better yet: weighted ERM with soft labels

The main problem with majority voting was its sensitivity to the decision of a single party. Let $\alpha(x)$ be the fraction of positive votes from M classifiers given a sample x:

$$\alpha(x) = \frac{1}{M}\sum_{j=1}^{M} I[h_j(x) = 1]. \quad (10)$$

In terms of α, the original loss $l(w^T x v(x))$ for majority voting can be written as

$$\begin{aligned} l(yw^T x) &= I[\alpha(x) \geq 0.5]\; l(w^T x) \\ &\quad + \; I[\alpha(x) < 0.5]\; l(-w^T x), \end{aligned} \quad (11)$$

which changes abruptly when the fraction $\alpha(x)$ crosses the boundary $\alpha = 0.5$. We remedy the situation by introducing the new *weighted* loss:

$$l^\alpha(\cdot) = \alpha(x) l(w^T x) + (1 - \alpha(x)) l(-w^T x). \quad (12)$$

The new loss has the following properties. When the M votes given a sample x are unanimously positive (or negative), then the weighted loss is $l^\alpha(\cdot) = l(w^T x)$ (or $l(-w^T x)$), same as the original loss. If the votes are almost evenly split between positive and negative, then the weighted loss is $l^\alpha(\cdot) \simeq 0.5\; l(w^T x) + 0.5\; l(-w^T x)$ which is insensitive to the change of label by a single vote, unlike the original loss. Specifically, a single vote can change $l^\alpha(\cdot)$ only by a factor of $1/M$ (see Proof of Theorem 3.)

We provide a natural interpretation of $\alpha(x)$ and the weighted loss in the following. For the purpose of analysis, assume that the local classifiers $h_1(x), ..., h_M(x)$ are from the same hypothesis class.[2] Since the local training data are i.i.d. samples from $P(x, y)$, the local classifiers $\{h_1(x), ..., h_M(x)\}$ can be considered random hypotheses, as in (Breiman, 1996a). Let $Q(j|x)$ be the probability of such a random hypothesis $h(x)$ predicting label j given x:

$$Q(j|x) = P(h(x) = j|x), \quad (13)$$

Then the fraction $\alpha(x) = \frac{1}{M}\sum_{j=1}^{M} I[h_j(x) = 1]$ is an unbiased estimate of $Q(1|x)$. Furthermore, the weighted loss is directly related to the unweighted loss:

Lemma 2. *For any w, the expectation of the weighted loss (12) is asymptotically the expectation of the unweighted loss:*

$$\lim_{M\to\infty} E_x[l^\alpha(w)] = E_{x,v}[l(w^T x v)]. \quad (14)$$

Proof. The expected risk $E_{x,v}[l(vw^T x)]$ is

$$\begin{aligned} &= E_x E_{v|x}[l(vw^T x)] \\ &= E_x[Q(1|x)l(w^T x) + Q(-1|x)l(-w^T x)] \\ &= E_x[\lim_{M\to\infty} \alpha(x) l(w^T x) + (1 - \lim_{M\to\infty} \alpha(x)) l(-w^T x)] \\ &\quad \text{(the law of large numbers)} \\ &= \lim_{M\to\infty} E_x[\alpha(x) l(w^T x) + (1 - \alpha(x)) l(-w^T x)] \\ &\quad \text{(bounded } \alpha \text{ and } l \text{ for } \forall x \in \mathcal{X}) \\ &= \lim_{M\to\infty} E_x[l^\alpha(w)]. \end{aligned} \quad (15)$$

□

This shows that minimizing the expected weighted loss is asymptotically the same as minimizing the standard expected loss, when the target v is a *probabilistic* concept from $P(h(x) = v)$ of the random hypothesis, as opposed to a deterministic concept $v(x)$ from majority voting.

The auxiliary dataset with 'soft' labels is now

$$S = \{(x_1, \alpha(x_1)),\; ...\; , (x_N, \alpha(x_N))\}. \quad (16)$$

where $x_i \in \mathcal{X}$ is an i.i.d. sample from the same distribution $P(x)$ as the private data, and $0 \leq \alpha \leq 1$. Note that we are not trying to learn a regression function $\mathcal{X} \to [0, 1]$ but to learn a classifier $\mathcal{X} \to \{-1, 1\}$ using α as a real-valued oracle on $P(y = 1|x)$. Consequently, we find a global classifier by minimizing the regularized *weighted* empirical risk

$$R_S^\lambda(w) = \frac{1}{N}\sum_{i=1}^{N} l^\alpha(h(x_i; w), \alpha_i) + \frac{\lambda}{2}\|w\|^2, \quad (17)$$

where $\alpha_i = \alpha(x_i)$. We use $R^\lambda(w)$ and $R(w)$ without the subscript S to denote expected weighted risks with and without regularization.

We again use output perturbation to make the classifier differentially private as summarized in Algorithm 2.

Algorithm 2 DP Ensemble by Weighted ERM

Input: $h_1, ..., h_M$ (local classifiers), X (auxiliary unlabeled samples), ϵ, λ
Output: w_p
Begin
  for $i = 1,\; ...\; , N$ **do**
    Compute $\alpha(x_i)$ by (10)
  end for
  Find the minimizer of w_s of (17) with $\{(x_i, \alpha(x_i))\}$
  Sample a random vector η from $p(\eta) \propto e^{-0.5M\lambda\epsilon\|\eta\|}$
  Output $w_p = w_s + \eta$

4.4. Privacy and performance

Compared to Theorem 1 for majority-voted ERM with a noise of $P(\eta) \propto e^{-\frac{\lambda\epsilon}{2}\|\eta\|}$, we have the following result:

[2] Our differential privacy guarantee holds whether they are from the same hypothesis class or not.

Theorem 3. *The perturbed output $w_p = w_s + \eta$ from Algorithm 2 with $p(\eta) \propto e^{-\frac{M\lambda\epsilon}{2}\|\eta\|}$ is ϵ-differentially private.*

That is, we now require $1/M$ times smaller noise to achieve the same ϵ-differential privacy. This directly impacts the performance of the corresponding global classifier as follows.

Theorem 4. *Let w_0 be any reference hypothesis. Then with probability of at least $1 - \delta_p - \delta_s$ over the privacy mechanism (δ_p) and over the choice of samples (δ_s),*

$$R(w_p) \leq R(w_0) + \frac{4d^2(c+\lambda)\log^2(d/\delta_p)}{\lambda^2 M^2 \epsilon^2} + \frac{16(32+\log(1/\delta_s))}{\lambda N} + \frac{\lambda}{2}\|w_0\|^2. \quad (18)$$

The generalization error bound above has the $O(M^{-2}\epsilon^{-2})$ term compared to the $O(\epsilon^{-2})$ term for majority-voted ERM (9). This implies that by choosing a large M, Algorithm 2 can find a solution whose expected risk is close to the minimum of a non-private solution for any fixed $\epsilon > 0$.

We remind the user that the results should be interpreted with a caution. The bounds in (9) and (18) indicate the goodness of private ERM solutions relative to the best non-private solutions with deterministic and probability concepts which are not the same task. Also, they do not indicate the goodness of the ensemble approach itself relative to a centrally-trained classifier using all private data without privacy consideration. We leave this comparison to empirical evaluation in the experiment section.

4.5. Extensions

We discuss extensions of Algorithms 1 and 2 to provide additional privacy for auxiliary data. More precisely, those algorithms can be made ϵ-differentially private for all private data of a single party and a single sample in the auxiliary data, by increasing the amount of perturbation as necessary. We outline the proof as follows. In the previous sections, a global classifier was trained on auxiliary data whose labels were generated either by majority voting or soft labeling. A change in the local classifier affects only the labels $\{v_i\}$ of the auxiliary data but not the features $\{x_i\}$. Now assume in addition that the feature of one sample from the auxiliary data can also change arbitrarily, i.e., $x_j \neq x'_j$ for some j and $x_i = x'_i$ for all $i \in \{1, ..., N\} \setminus \{j\}$. The sensitivity of the resultant risk minimizer can be computed similarly to the proofs of Theorems 1 and 3 in Appendix A of the supplementary material. Briefly, the sensitivity is upper-bounded by the absolute sum of the difference of gradients

$$\|\nabla g(w)\| \leq \frac{1}{N}\sum_{i=1}^{N}\|\nabla l(y_i w^T x_i) - \nabla l(y'_i w^T x'_i)\|. \quad (19)$$

For majority voting, one term in the sum (19) is

$$\|v(x_j)x_j l'(v(x_j)w^T x_j) - v'(x'_j)x'_j l'(v'(x'_j)w^T x'_j)\| \quad (20)$$

which is at most 2 for any $x_j, x'_j \in \mathcal{X}$, and therefore the sensitivity is the same whether $x_j = x'_j$ or not. As a result, Algorithm 1 is already ϵ-differentially private for both labeled and auxiliary data without modification. Furthermore, the privacy guarantee remains the same if we allow $x_j \neq x'_j$ for any number of samples. For soft labeling, one term in the sum (19) is

$$\|\alpha_j x_j l'(w^T x_j) - (1-\alpha_j)x_j l'(-w^T x_j) - \alpha'_j x'_j l'(w^T x'_j) + (1-\alpha'_j)x'_j l'(-w^T x'_j)\| \quad (21)$$

which is also at most 2 for any $x_j, x'_j \in \mathcal{X}$ and $\frac{2}{M}$ when $x_j = x'_j$. When only a single auxiliary sample changes, i.e., $x_j \neq x'_j$ for one j, the overall sensitivity increases by a factor of $\frac{N+M-1}{N}$. By increasing the noise by this factor, Algorithm 2 is ϵ-differentially private for both labeled and auxiliary data. Note that this factor $\frac{N+M-1}{N}$ can be bounded close to 1 if we increase the number of auxiliary samples N relative to the number of parties M.

5. Related work

To preserve privacy in data publishing, several approaches such as k-anonymity (Sweeney, 2002) and secure multiparty computation (Yao, 1982) have been proposed (see (Fung et al., 2010) for a review.) Recently, differential privacy (Dwork & Nissim, 2004; Dwork et al., 2006; Dwork, 2006) has addressed several weaknesses of k-anonymity (Ganta et al., 2008), and gained popularity as a quantifiable measure of privacy risk. The measure provides a bound on the privacy loss regardless of any additional information an adversary might have. Differential privacy has been used for a privacy-preserving data analysis platform (McSherry, 2009), and for sanitization of learned model parameters from a standard ERM (Chaudhuri et al., 2011). This paper adopts output perturbation techniques from the latter to sanitize non-standard ERM solutions from multiparty settings.

Private learning from multiparty data has been studied previously. In particular, several differentially-private algorithms were proposed, including parameter averaging through secure multiparty computation (Pathak et al., 2010), and private exchange of gradient information to minimize empirical risks incrementally (Rajkumar & Agarwal, 2012; Hamm et al., 2015). Our paper is motivated by (Pathak et al., 2010) but uses a very different approach to aggregate local classifiers. In particular, we use an ensemble approach and average the classifier decisions (Breiman, 1996a) instead of parameters, which makes our approach applicable to arbitrary and mixed classifier types. Advan-

tages of ensemble approaches in general have been analyzed previously, in terms of bias-variance decomposition (Breiman, 1996b), and in terms of the margin of training samples (Schapire et al., 1998).

Furthermore, we are using unlabeled data to augment labeled data during training, which can be considered a semi-supervised learning method (Chapelle et al., 2006). There are several related papers in this direction. Augmenting private data with non-private *labeled* data to lower the sensitivity of the output is straightforward, and was demonstrated in medical applications (Ji et al., 2014). Using non-private *unlabeled* data, which is more general than using labeled data, was demonstrated specifically to assist learning of random forests (Jagannathan et al., 2013). Our use of auxiliary data is not specific to classifier types. Furthermore, we present an extension to preserve the privacy of auxiliary data as well.

6. Experiments

We use three real-world datasets to compare the performance of the following algorithms:

- *batch*: classifier trained using all data ignoring privacy
- *soft*: private ensemble using soft-labels (Algorithm 2)
- *avg*: parameter averaging (Pathak et al., 2010)
- *vote*: private ensemble using majority voting (Algorithm 1)
- *indiv*: individually trained classifier using local data

We can expect *batch* to perform better than any private algorithm since it uses all private data for training ignoring privacy. In contrast, *indiv* uses only the local data for training and will perform significantly worse than *batch*, but it achieves a perfect privacy as long as the trained classifiers are kept local to the parties. We are interested in the range of ϵ where private algorithms (*soft*, *avg*, and *vote*) perform better than the baseline *indiv*.

To compare all algorithms fairly, we use only a single type of classier – binary or multiclass logistic regression. For Algorithms 1 and 2, both local and global classifiers are of this type as well. The only hyperparameter of the model is the regularization coefficient λ which we fixed to 10^{-4} after performing some preliminary experiments. About 10% of the original training data are used as auxiliary unlabeled data, and the rest 90% are randomly distributed to M parties as private data. We report the mean and s.d. over 10 trials for non-private algorithms and 100-trials for private algorithms.

6.1. Activity recognition using accelerometer

Consider a scenario where wearable device users want to train a motion-based activity classifier without revealing her data to others. To test the algorithms, we use the UCI Human Activity Recognition Dataset (Anguita et al., 2012), which is a collection of motion sensor data on a smart device by multiple subjects performing 6 activities (*walking*, *walking upstairs*, *walking downstairs*, *sitting*, *standing*, *laying*). Various time and frequency domain variables are extracted from the signal, and we apply PCA to get $d = 50$ dimensional features. The training and testing samples are $7K$ and $3K$, respectively.

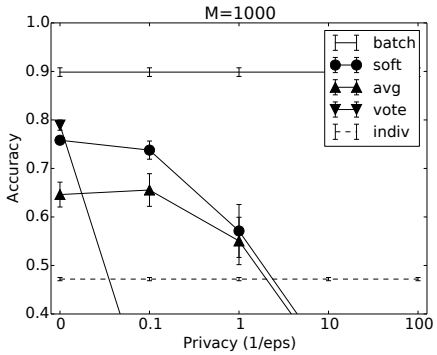

Figure 2. Test accuracy of private and non-private algorithms for activity recognition with $K = 6$ activity types.

We simulate a case with $M = 1K$ users (i.e., parties). Each user can use only 6 samples to train a local classifier. The remaining $1K$ samples are used as auxiliary unlabeled data. Figure 2 shows the test accuracy of using different algorithms with varying privacy levels. For non-private algorithms, the top solid line (*batch*) shows the accuracy of a batch-trained classifier at around 0.90, and the bottom dashed line (*indiv*) shows the averaged accuracy of local classifiers at around 0.47. At $1/\epsilon = 0$, the private algorithms achieve test accuracy of 0.79 (*vote*), 0.76 (*soft*) and 0.67 (*avg*), and as the the privacy level $1/\epsilon$ increases, the performance drops for all private algorithms. As expected from the bound (9), *vote* becomes useless even at $1/\epsilon = 0.1$, while *soft* and *avg* are better than *indiv* until $1/\epsilon = 1$. We fixed M to 1000 in this experiment due to the limited number of samples, the tendency in the graph is similar to other experiments with larger M's.

6.2. Network intrusion detection

Consider a scenario where multiple gateways or routers collect suspicious network activities independently, and aim to collaboratively build an accurate network intrusion detector without revealing local traffic data. For this task we use the KDD-99 dataset, which consists of examples of 'bad' connections, called intrusions or attacks, and 'good' normal connections. Features of this dataset consists of continuous values and categorical attributes. To apply logistic regression, we change categorical attributes to one-

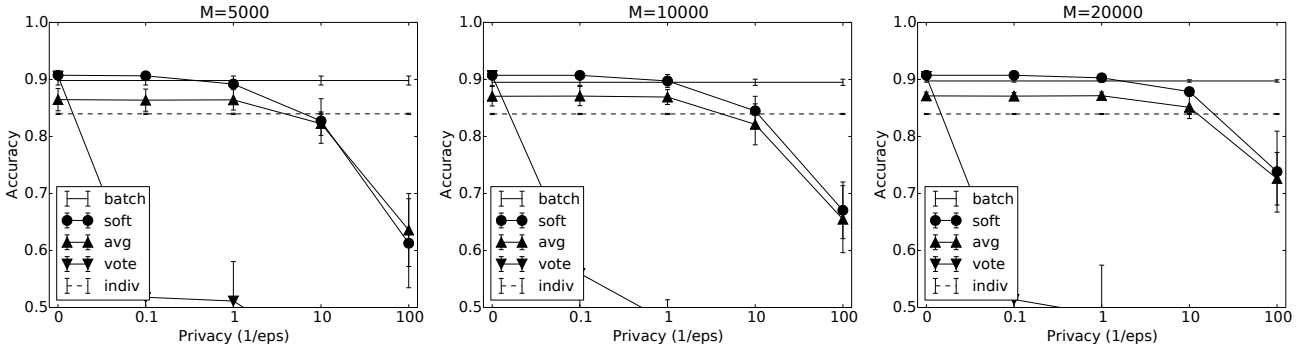

Figure 3. Test accuracy of private and non-private algorithms for network intrusion detection.

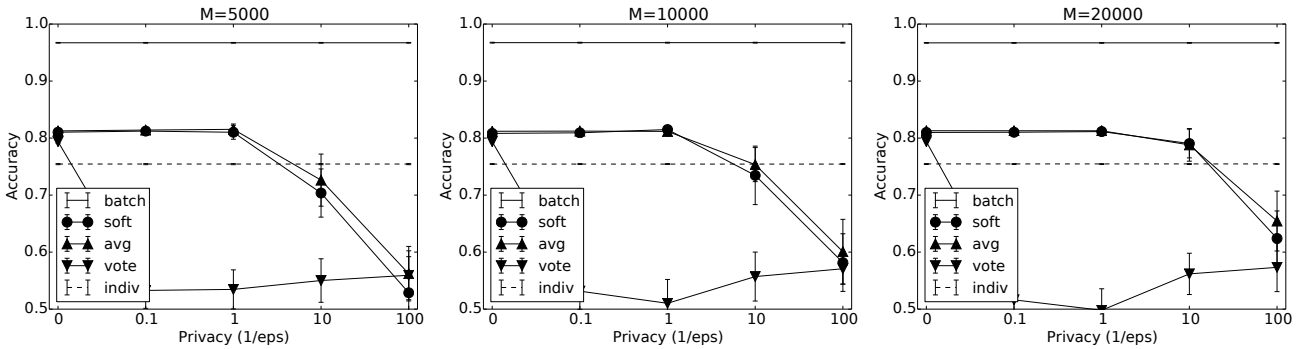

Figure 4. Test accuracy of private and non-private algorithms for malicious URL detection.

hot vectors to get $d = 123$ dimensional features. The training and testing samples are $493K$ and $311K$, respectively.

We simulate cases with $M = 5K/10K/20K$ parties. Each party can only use 22 samples to train a local classifier. The remaining $43K$ samples are used as auxiliary unlabeled data. Figure 3 shows the test accuracy of using different algorithms with varying privacy levels. For each of $M = 5K/10K/20K$ the tendency of algorithms is similar to Figure 2 – for a small $1/\epsilon$, private algorithms perform roughly in between *batch* and *indiv*, and as $1/\epsilon$ increases private algorithms start to perform worse than *indiv*. When M is large (e.g., $M = 20K$), private algorithms *soft* and *avg* hold their accuracy better than when M is small (e.g., $M = 5K$.) In particular, *soft* performs nearly as well as the non-private *batch* until around $1/\epsilon = 10$ compared to avg.

6.3. Malicious URL prediction

In addition to network intrusion detection, multiple parties such as computer users can collaborate on detecting malicious URLs without revealing the visited URLs. The Malicious URL Dataset (Ma et al., 2009) is a collection of examples of malicious URLs from a large Web mail provider. The task is to predict whether a URL is malicious or not by various lexical and host-based features of the URL. We apply PCA to get $d = 50$ dimensional feature vectors. We choose days 0 to 9 for training, and days 10 to 19 for testing, which amount to $200K$ samples for training and $200K$ samples for testing.

We simulate cases with $M = 5K/10K/20K$ parties. Each party can use only 9 samples to train a local classifier. The remaining $16K$ samples are used as auxiliary unlabeled data. Figure 4 shows the test accuracy of using different algorithms with varying privacy levels. The gap between *batch* and other algorithms is larger compared to the previous experiment (Figure 3), most likely due to the smaller number (=9) of samples per party. However, the overall tendency is very similar to previous experiments.

7. Conclusion

In this paper, we propose a method of building global differentially private classifiers from local classifiers using two new ideas: 1) leveraging unlabeled auxiliary data to transfer the knowledge of the ensemble, and 2) solving a weighted ERM using class probability estimates from the ensemble. In general, privacy comes with a loss of classification performance. We present a solution to minimize the performance gap between private and non-private ensembles demonstrated with real world tasks.

Acknowledgements

This research was supported in part by funding from NSF IIS EAGER 1550757 and Google Faculty Research Award 2015.

References

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J.L. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *Ambient assisted living and home care*, pp. 216–223. Springer, 2012.

Breiman, L. Bagging predictors. *Machine learning*, 24(2):123–140, 1996a.

Breiman, L. Bias, variance, and arcing classifiers. Technical report, Statistics Department, University of California, Berkeley, CA, USA, 1996b.

Chapelle, O., Schölkopf, B., and Zien, A. (eds.). *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006. URL http://www.kyb.tuebingen.mpg.de/ssl-book.

Chaudhuri, K., Monteleoni, C., and Sarwate, A.D. Differentially private empirical risk minimization. *The Journal of Machine Learning Research*, 12:1069–1109, 2011.

Dwork, C. Differential privacy. In *Automata, languages and programming*, pp. 1–12. Springer, 2006.

Dwork, C. and Nissim, K. Privacy-preserving datamining on vertically partitioned databases. In *Advances in Cryptology–CRYPTO 2004*, pp. 528–544. Springer, 2004.

Dwork, C. and Roth, A. The algorithmic foundations of differential privacy. *Theoretical Computer Science*, 9(3-4):211–407, 2013.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography*, pp. 265–284. Springer, 2006.

Fung, B., Wang, K., Chen, R., and Yu, P.S. Privacy-preserving data publishing: A survey of recent developments. *ACM Comp. Surveys (CSUR)*, 42(4):14, 2010.

Ganta, S.R., Kasiviswanathan, S.P., and Smith, A. Composition attacks and auxiliary information in data privacy. In *Proc. ACM SIGKDD*, pp. 265–273. ACM, 2008.

Hamm, J., Champion, A., Chen, G., Belkin, M., and Xuan, D. Crowd-ML: A privacy-preserving learning framework for a crowd of smart devices. In *Proceedings of the 35th IEEE International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2015.

Jagannathan, G., Monteleoni, C., and Pillaipakkamnatt, K. A semi-supervised learning approach to differential privacy. In *Data Mining Workshops (ICDMW), 2013 IEEE 13th International Conference on*, pp. 841–848. IEEE, 2013.

Ji, Z., Jiang, X., Wang, S., Xiong, L., and Ohno-Machado, L. Differentially private distributed logistic regression using private and public data. *BMC medical genomics*, 7(Suppl 1):S14, 2014.

Ma, J., Saul, L.K., Savage, S., and Voelker, G.M. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 681–688. ACM, 2009.

McSherry, F.D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pp. 19–30. ACM, 2009.

Pathak, M., Rane, S., and Raj, B. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Information Processing Systems*, pp. 1876–1884, 2010.

Politis, D. N., Romano, J. P., and Wolf, M. *Subsampling*. Springer Verlag, 1999.

Rajkumar, A. and Agarwal, S. A differentially private stochastic gradient descent algorithm for multiparty classification. In *International Conference on Artificial Intelligence and Statistics*, pp. 933–941, 2012.

Schapire, R.E., Freund, Y., Bartlett, P., and Lee, W.S. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of statistics*, pp. 1651–1686, 1998.

Sweeney, L. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

Yao, A.C. Protocols for secure computations. In *2013 IEEE Symp. Found. Comp. Sci.*, pp. 160–164. IEEE, 1982.