A Dual Perturbation Approach for Differential Private ADMM-Based Distributed Empirical Risk Minimization

Tao Zhang
Department of Electrical and Computer Engineering
Tandon School of Engineering
New York University
Brooklyn, NY, 11201
tz636@nyu.edu

Quanyan Zhu
Department of Electrical and Computer Engineering
Tandon School of Engineering
New York University
Brooklyn, NY, 11201
qz494@nyu.edu

ABSTRACT

The rapid growth of data has raised the importance of privacy-preserving techniques in distributed machine learning. In this paper, we develop a privacy-preserving method to a class of regularized empirical risk minimization (ERM) machine learning problems. We first decentralize the learning algorithm using the alternating direction method of multipliers (ADMM), and propose the method of *dual variable perturbation* to provide dynamic differential privacy. The mechanism leads to a privacy-preserving algorithm under mild conditions of the convexity and differentiability of the loss function and the regularizer. We study the performance of the algorithm measured by the number of data points required to achieve a bounded error. To design an optimal privacy mechanism, we analyze the fundamental tradeoff between privacy and accuracy, and provide guidelines to choose privacy parameters. Numerical experiments using the real-world database are performed to corroborate the results on the privacy and utility tradeoffs and design.

Keywords

Machine Learning; Distributed Optimization; Differential Privacy; ADMM; Privacy Tradeoffs

1 Introduction

Recent years have witnessed that *Distributed machine learning* is a promising way to manage the deluge of data. The size of the training data ranges from $1TB$ to $1PB$ [13]; as a result, a centralized machine learning that collects and processes the data can lead to significant computational complexity and communications overhead. Therefore, a decentralized approach to machine learning is essential to reduce the computational cost, provide the scalability of the data processing and improve the quality of decision-making.

Alternating direction method of multiplier (ADMM) is one suitable approach to decentralizing a centralized machine learning problem. ADMM enables distributed training over a network of collaborative nodes which exchange their parameters and outcomes with the neighbors. However, serious privacy concerns arise from the communications between two neighboring nodes, which process sensitive data including social network data, the web search histories, financial information, and medical records. It is possible for an adversary to acquire confidential information about the training data of individual nodes by observing the outcome of the learning. The adversary can be either a member of the learning network or an outsider. Differential privacy is a well-suitable concept that provides a strong privacy guarantee that the absence of a single database item does not allow an adversary to distinguish (substantially) an individual data point [8].

This paper focuses on a class of distributed ADMM-based *empirical risk minimization* (ERM) problems, and develops a randomized algorithm that can provide differential privacy [8, 17] while keeping the learning procedure accurate. The privacy concepts of [8, 17] is extended to distributed machine learning over networks based on ADMM, and we propose a privacy-preserving scheme of the regularized ERM-based optimization. The method is *dual variable perturbation* (DVP), in which we perturb the dual variable of each node at every ADMM iteration.

We investigate the performance of the algorithms and show that the DVP is useful for non-separable learning problems. We characterize the fundamental tradeoffs between privacy and accuracy by formulating an optimization problem and use numerical experiments to demonstrate the optimal design of privacy mechanisms. We use ADMM to decentralize regularized ERM algorithms to achieve distributed training of large datasets. Dynamic differential privacy is guaranteed for the distributed algorithm using the DVP, which adds noise to the update of the dual variable. We provide the theoretical performance guarantees of the DVP version of the distributed ERM with l_2 regularization. The performance is measured by the number of sample data points required to achieve certain criteria. We also propose a design principle to select the optimal privacy parameters by solving an optimization problem.

1.1 Related Work

A significant amount of research has investigated the distributed classification learning algorithm. These works have focused on either enhancing the efficiency of the learning model or on producing a global classifier from multiple distributed local classifier trained at individual nodes. Efforts

AISec'16, October 28, 2016, Vienna, Austria

DOI: http://dx.doi.org/10.1145/2996758.2996762

have been on making the distributed algorithm suitable for large-scale datasets, e.g., MapReduce has been used to explore the performance improvements [7]. In addition, methods such as voting classification [5], mixing parameters [14], and ADMM methods [10], have been used to achieve distributed computation. ADMM is used in our approach to distributed machine learning, in which the centralized problem acts as a group of distributed convex optimization subproblems connected by the consensus constraints on the decision parameters over a network.

In privacy-preserving data mining research, a large amount of literature on data perturbation for privacy (e.g., [9],[12]) has focused on additive or multiplicative perturbations of individual samples, which might affect certain relationships among different samples in the database. Many works also have studied the differential-private machine learning. For example, Kasiviswanathan et al. have derived a general method for probably approximately correct (PAC, [19]) in [11]. A body of existing literature has investigated the tradeoff privacy and accuracy while researching on the theory of differential privacy (examples include [8, 15, 2]). This work extends the notion of differential privacy to a dynamic setting, and defines dynamic differential privacy to capture the distributed and iterative nature of the ADMM-based distributed ERM.

1.2 Organization of the Paper

The rest of the paper is organized as follows. In Section 2, we present the ADMM approach to decentralizing a centralized ERM problem, and describe the privacy concerns associated with the distributed machine learning. Section 3 presents the DVP algorithm to provide dynamic differential privacy. Section 4 studies the performance of the DVP algorithm and Section 5 shows numerical experiments to corroborate the results and optimal design principles to the tradeoff between privacy and accuracy. Finally, Section 6 presents concluding remarks and future research directions.

2 Problem Statement

Consider a connected network, which contains P nodes described by an undirected graph $G(\mathcal{P},\mathcal{E})$ with the set of nodes $\mathcal{P}=\{1,2,3,...,P\}$, and a set of edges $\mathcal{E}$ denoting the links between connected nodes. A particular node $p\in\mathcal{P}$ only exchanges information between its neighboring node $j\in\mathcal{N}_p$, where $j\in\mathcal{N}_p$ is the set of all neighboring nodes of node p, and $N_p=|\mathcal{N}_p|$ is the number of neighboring nodes of node p. Each node p contains a dataset $D_p=\{(x_{ip},y_{ip})\subset X\times Y : i=0,1,...,B_p\}$, which is of size B_p with data vector $x_{ip}\in X\subseteq\mathbb{R}^d$, and the corresponding label $y_{ip}\in Y:=\{-1,1\}$. The entire network therefore has a set of data $\hat{D}=\bigcup_{p\in\mathcal{P}}D_p$.

The target of the centralized classification algorithm is to find a classifier $f:X\rightarrow Y$ using all available data $\hat{D}$ that enables the entire network to classify any data x' input to a label $y'\in\{-1,1\}$. Let $Z_{C_1}(f|\hat{D})$ be the objective function of a regularized empirical risk minimization problem (CR-ERM), defined as follows:

$$Z_{C_1}(f|\hat{D}):=\frac{C^R}{B_p}\sum_{p=1}^{P}\sum_{i=1}^{B_p}\hat{\mathcal{L}}(y_{ip},\ f^Tx_{ip})+\rho R(f), \tag{1}$$

where $C^R\leqslant B_p$ is a regularization parameter, and $\rho>0$ is the parameter that controls the impact of the regularizer. Suppose that $\hat{D}$ is available to the fusion center node, then we can choose the global classifier $f:X\rightarrow Y$ that minimizes the CR-ERM.

The *loss function* $\hat{\mathcal{L}}(y_{ip},\ f^Tx_{ip}):\mathbb{R}\times\mathbb{R}^d\times\mathbb{R}^d\rightarrow\mathbb{R}$, is used to measure the quality of the classifier trained. In this paper, we focus on the specific loss function $\hat{\mathcal{L}}(y_{ip},\ f^Tx_{ip})=\mathcal{L}(y_{ip}f^Tx_{ip})$. The function $R(f)$ in (1) is a regularizer that prevents overfitting. In this paper, we have the following assumptions on the loss, regularization functions, and the data.

Assumption 1. The loss function $\mathcal{L}$ is strictly convex and doubly differentiable of f with $|\mathcal{L}'|\leqslant 1$ and $|\mathcal{L}''|\leqslant c_1$, where c_1 is a constant. Both $\mathcal{L}$ and $\mathcal{L}'$ are continuous.

Assumption 2. The regularizer function $R(\cdot)$ is continuous, differentiable and 1-strongly convex. Both $R(\cdot)$ and $\nabla R(\cdot)$ are continuous.

Assumption 3. We assume that $\|x_{ip}\|\leqslant 1$. Since $y_{ip}\in\{-1,1\}$, $|y_{ip}|=1$.

2.1 Distributed ERM

To decentralize CR-ERM, we introduce decision variables $\{f_p\}_{p=1}^P$, where node p determines its own classifier f_p, and impose consensus constraints $f_1=f_2=...=f_P$ that guarantee global consistency of the classifiers. Let $\{w_{jp}\}$ be the auxiliary variables to decouple f_p of node p from its neighbors $j\in\mathcal{N}_p$. Then, the consensus-based reformulation of (1) becomes

$$\begin{aligned}\min_{\{f_p\}_{p=1}^P}\quad & Z_{C_2}:=\frac{C^R}{B_p}\sum_{p=1}^{P}\sum_{i=1}^{B_p}\mathcal{L}(y_{ip}f_p^Tx_{ip})+\sum_{p=1}^{P}\rho R(f_p).\\ \text{s.t.}\quad & f_p=w_{pj},\ w_{pj}=f_j, p=1,...,P, j\in\mathcal{N}_p\end{aligned} \tag{2}$$

where $Z_{C_2}(\{f_p\}_{p\in\mathcal{P}}|\hat{D})$ is the reformulated objective as a function of $\{f_p\}_{p=1}^P$. According to Lemma 1 in [10], if $\{f_p\}_{p=1}^P$ presents a feasible solution of (2) and the network is connected, then problems (1) and (2) are equivalent, i.e., $f=f_p, p=1,...,P$, where f is a feasible solution of CR-ERM. Problem (2) can be solved in a distributed fashion using the alternative direction method of multiplier (ADMM) with each node $p\in\mathcal{P}$ optimizing the following distributed regularized empirical risk minimization problem (DR-ERM):

$$Z_p(f_p|D_p):=\frac{C^R}{B_p}\sum_{i=1}^{B_p}\mathcal{L}(y_{ip}f_p^Tx_{ip})+\rho R(f_p). \tag{3}$$

The augmented Lagrange function associated with the DR-ERM is:

$$\begin{aligned}&L_p^D(f_p,w_{pj},\lambda_{pj}^k)\\ &=Z_p+\sum_{i\in\mathcal{N}_p}\left(\lambda_{pi}^a\right)^T(f_p-w_{pi})+\sum_{i\in\mathcal{N}_p}\left(\lambda_{pi}^b\right)^T(w_{pi}-f_i)\\ &+\frac{\eta}{2}\sum_{i\in\mathcal{N}_p}(\parallel f_p-w_{pi}\parallel^2+\parallel w_{pi}-f_i\parallel^2).\end{aligned} \tag{4}$$

The distributed iterations solving (3) are:

$$f_p(t+1)=\arg\min_{f_p}L_p^D\big(f_p,w_{pj}(t),\lambda_{pj}^k(t)\big), \tag{5}$$

$$w_{pj}(t+1)=\arg\min_{w_{pj}}L_p^D\big(f_p(t+1),w_{pj},\lambda_{pj}^k(t)\big), \tag{6}$$

Algorithm 1 Distributed ERM

```
Required: Randomly initialize $f_p, \lambda_p = \mathbf{0}_{d\times 1}$ for every $p \in \mathcal{P}$
Input: $\hat{D}$
for $t = 0, 1, 2, 3, ...$ do
  for $p = 0$ to $P$ do
    Compute $f_p(t+1)$ via (10).
  end for
  for $p = 0$ to $P$ do
    Broadcast $f_p(t+1)$ to all neighbors $j \in \mathcal{N}_p$.
  end for
  for $p = 0$ to $P$ do
    Compute $\lambda_p(t+1)$ via (11).
  end for
end for
Output: $f^*$.
```

$$\lambda^a_{pj}(t+1) = \lambda^a_{pj}(t) + \eta(f_p(t+1) - w_{pj}(t+1)),\quad p \in \mathcal{P},\ j \in \mathcal{N}_p, \tag{7}$$

$$\lambda^b_{pj}(t+1) = \lambda^b_{pj}(t) + \eta(w_{pj}(t+1) - f_p(t+1)),\quad p \in \mathcal{P},\ j \in \mathcal{N}_p. \tag{8}$$

According to Lemma 2 in [10], iterations (5) to (8) can be further simplified by initializing the dual variables $\lambda^k_{pj} = \mathbf{0}_{d\times d}$, and letting $\lambda_p(t) = \sum_{j\in\mathcal{N}_p} \lambda^k_{pj}$, $p \in \mathcal{P}$, $j \in \mathcal{N}_p$, $k = a$, b, we can combine (7) and (8) into one update. Thus, we simplify (5)-(8) by introducing the following: Let $L^N_p(t)$ be the short-hand notation of $L^N_p(\{f_p\}, \{f_p(t)\}, \{\lambda_p(t)\})$ as :

$$\begin{aligned} L^N_p(t) :=& \frac{C^R}{B_p}\sum_{i=1}^{B_p} \mathcal{L}(y_{ip} f_p^T x_{ip}) + \rho R(f_p) + 2\lambda_p(t)^T f_p \\ &+ \eta \sum_{i\in\mathcal{N}_p} \parallel f_p - \frac{1}{2}(f_p(t) + f_i(t)) \parallel^2 . \end{aligned} \tag{9}$$

The ADMM iterations (5)-(8) can be reduced to

$$f_p(t+1) = \arg\min_{f_p} L^N_p(f_p, f_p(t), \lambda_p(t)), \tag{10}$$

$$\lambda_p(t+1) = \lambda_p(t) + \frac{\eta}{2}\sum_{j\in\mathcal{N}_p}[f_p(t+1) - f_j(t+1)]. \tag{11}$$

The ADMM-based distributed ERM iterations (10)-(11) are summarized in Algorithm 1. Every node $p \in \mathcal{P}$ updates its local $d \times 1$ estimates $f_p(t)$ and $\lambda_p(t)$. At iteration $t+1$, node p updates the local $f_p(t+1)$ through (10). Next, node p broadcasts the latest $f_p(t+1)$ to all its neighboring nodes $j \in \mathcal{N}_p$. Iteration $t+1$ finishes as each node updates the $\lambda_p(t+1)$ via (11).

Every iteration of our algorithm is still a minimization problem similar to the centralized problem (1). However, the number of variables participating in solving (10) per node per iteration is N_p, which is much smaller than the one in the centralized problem, which is $\sum_{p=1}^{P} N_p$. There are several methods to solve (10). For instance, projected gradient method, Newton method, and Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [6] that approximates the Newton method, to name a few.

ADMM-based distributed machine learning has benefits due to its high scalability. It also provides a certain level of privacy since nodes do not communicate data directly but their decision variable f_p. However, the privacy arises when an adversary can make intelligent inferences at each step and extract the sensitive information based on his observation of the learning output of his neighboring nodes. Simple anonymization is not sufficient to address this issue as discussed in Section 1. In the following subsection, we will discuss the adversary models, and present differential privacy solutions.

2.2 Privacy Concerns

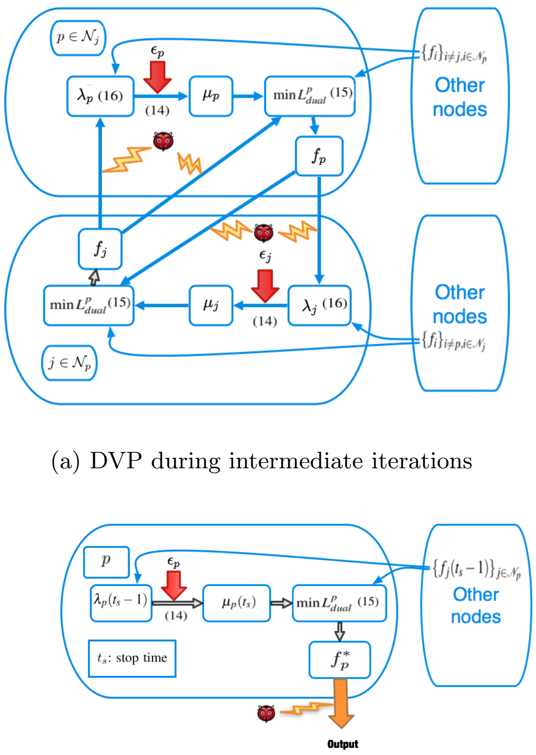
Although the data stored at each node is not exchanged during the entire ADMM algorithm, the potential privacy risk still exists. Suppose that the dataset D_p stored at node p contains sensitive information in data point (x_i, y_i) that is not allowed to be released to other nodes in the network or anyone else outside. Let $K : \mathbb{R}^d \to \mathbb{R}$ be the randomized version of Algorithm 1, and let $\{f^*_p\}_{p\in\mathcal{P}}$ be the output of K at all the nodes. Then, the output $\{f^*_p\}_{p\in\mathcal{P}}$ is random. In the distributed version of the algorithm, each node optimizes its local empirical risk based on its own dataset D_p. Let K^t_p be the node-p-dependent stochastic sub-algorithm of K at iteration t, and let $f_p(t)$ be the output of $K^t_p(D_p)$ at iteration t inputting D_p. Hence, the output $f_p(t)$ is stochastic at each t. In this work, we consider the following attack model. The adversary can access the learning outputs of intermediate ADMM iterations as well as the final output. This type of adversary aims to obtain sensitive information about the private data point of the training dataset by observing the output $f_p(t)$ of K^t_p or f^*_p of K for all $p \in \mathcal{P}$ at every stage t of the training. We protect the privacy of distributed network using the definition of *differential privacy* in [8]. Specifically, we require that a change of any single data point in the dataset might only change the distribution of the output of the algorithm slightly, which is visible to the adversary; this is done by adding randomness to the output of the algorithm. Let D_p and D'_p be two datasets differing in one data point; i.e., let $(x_{ip}, y_{ip}) \subset D_p$, and $(x'_{ip}, y'_{ip}) \subset D'_p$, then $(x_{ip}, y_{ip}) \neq (x'_{ip}, y'_{ip})$. In other words, their *Hamming Distance*, which is defined as $H_d(D_p, D'_p) = \sum_{i=0}^{B_p} \mathbf{1}\{i : x_i \neq x'_i\}$, equals 1; i.e., $H_d(D_p, D'_p) = 1$.

To protect the privacy against the adversary, we propose the concept of dynamic differential privacy, which enables the dynamic algorithm to be privacy-preserving at every stage of the learning.

Definition 1. *(Dynamic $\alpha(t)$-Differential Privacy (DDP)) Consider a network of P nodes $\mathcal{P} = \{1, 2, ..., P\}$, and each node p has a training dataset D_p, and $\hat{D} = \bigcup_{p\in\mathcal{P}} D_p$. Let $K : \mathbb{R}^d \to \mathbb{R}$ be a randomized version of Algorithm 1. Let $\alpha(t) = (\alpha_1(t), \alpha_2(t), ..., \alpha_P(t)) \in \mathbb{R}^P_+$, where $\alpha_p(t) \in \mathbb{R}_+$ is the privacy parameter of node p at iteration t. Let K^t_p be the node-p-dependent sub-algorithm of K, which corresponds to an ADMM iteration at t that outputs $f_p(t)$. Let D'_p be any dataset with $H_d(D'_p, D_p) = 1$, and $g_p(t) = K^t_p(D'_p)$. We say that the algorithm K is* dynamic $\alpha_p(t)$-differential private *(DDP) if for any dataset D'_p, and for all $p \in \mathcal{P}$ that can be observed by an adversary, and for all possible sets of the outcomes $S \subseteq \mathbb{R}$, the following inequality holds:*

$$\Pr[f_p(t) \in S] \leqslant e^{\alpha_p(t)} \cdot \Pr[g_p(t) \in S], \tag{12}$$

for all $t \in \mathbb{Z}$ during a learning process. The probability is taken with respect to $f_p(t)$, the output of K^t_p at every stage t. The algorithm K is called dynamic $\alpha(t)$-differential private *if the above conditions are satisfied.*



(a) DVP during intermediate iterations

(b) The final iteration of DVP

Figure 1. Illustration of DVP: (a) DVP during intermediate iterations. The perturbed μ_p participates in the (15). As a result, the output f_p at each iteration is a random variable, and the transmission of f_p is differential private. (b) The final iteration of DVP. Note that the evil face symbol represents the adversary.

Definition 1 provides a suitable differential privacy concept for the adversary. For dynamic $\alpha_p(t)$-differential private algorithms, the adversaries cannot extract additional information by observing the intermediate updates of $f_p(t)$ at each step. Clearly, the algorithm with ADMM iterations shown in (10) to (11) is not dynamic $\alpha_p(t)$-differential private. This is because the intermediate and final optimal output f_p's are deterministic given dataset D_p. For D'_p with $H_d(D_p, D'_p) = 1$, the classifier will change completely, and the probability density $\Pr([f_p|D'_p]) = 0$, which leads to the ratio of probabilities $\frac{\Pr[f_p|D_p]}{\Pr[f_p|D'_p]} \to \infty$. Please note that the optimization at each iteration in ADMM is independent of each other different iteration. This property of ADMM makes it possible that the privacy at each node each iteration is independent; the level of privacy at node $p \in \mathcal{P}$ iteration t totally depends on the value of $\alpha_p(t)$ chosen at time t and is independent of $\alpha_j(t')$, for all $t' \neq t$ and all $j \neq p$. Thus, our dynamic privacy is also independent of the number of iterations. As a result, the adversaries cannot obtain additional information from each iteration and there is nothing in previous iterations they can take advantage of to extract more information in later iterations.

3 Dynamic Private Preserving

In this subsection, we describe the algorithm that provides dynamic α-differential privacy defined in Section 2.2. We extend the definition of *differential privacy* in [8], in which the output, i.e., the variable in our case, is perturbed by random noise before releasing it. The fact that the optimization at each iteration in ADMM is independent of each other at different iterations has made it possible to deal with the privacy issue at a node $p \in \mathcal{P}$ in each iteration individually. However, the directly perturbed variables will be transmitted to neighboring nodes and thus the noise terms from all the neighboring nodes as well as the node p itself will be directly involved in the optimization at the next iteration. Thus, we perturb the Lagrange multiplier, i.e., dual variable, instead. The perturbed dual variable randomizes optimization at each iteration, and thus the output of each iteration. We name this method as the dual variable perturbation (DVP).

3.1 Dual Variable Perturbation

In the DVP, the dual variables $\{\lambda_p(t)\}_{p=1}^{P}$ are perturbed with a random noise vector $\epsilon_p(t) \in \mathbb{R}^d$ with the probability density function $\mathcal{K}_p(\epsilon) \sim e^{-\zeta_p(t)\|\epsilon\|}$, where $\zeta_p(t)$ is a parameter related to the value of $\alpha_p(t)$, and $\| \cdot \|$ denotes the l_2 norm. At each iteration, we first perturb the dual variable $\lambda_p(t)$, obtained from the last iteration, and store it in a new variable $\mu_p(t) = \lambda_p(t) + \epsilon_p(t)$. Now the corresponding node-p-based augmented Lagrange function $L_p^N(t)$ becomes

$$L_p^{dual}\big(f_p, f_p(t), \mu_p(t+1), \{f_i(t)\}_{i \in \mathcal{N}_p}\big),$$

defined as follows, and $L_p^{dual}(t)$ is used as a short-hand notation:

$$\begin{aligned} L_p^{dual}(t) = \quad & \tfrac{C^R}{B_p} \textstyle\sum_{i=1}^{B_p} \mathcal{L}(y_{ip} f_p^T x_{ip}) + \rho R(f_p) \\ & +2\mu_p(t+1)^T f_p + \tfrac{\Phi}{2} \parallel f_p \parallel^2 \\ & +\eta \textstyle\sum_{i \in \mathcal{N}_p} \parallel f_p - \tfrac{1}{2}(f_p(t) + f_i(t)) \parallel^2, \end{aligned} \tag{13}$$

where $\frac{\Phi}{2} \parallel f_p \parallel^2$ is an additional penalty. As a result, the minimizer of $L_p^{dual}(t)$ is random. At each iteration, we first perturb the dual variable $\lambda_p(t)$, obtained from the last iteration, and store it in a new variable $\mu_p(t+1)$.

Now, the iterations (10)-(11) become follows:

$$\mu_p(t+1) = \lambda_p(t) + \frac{C^R}{2B_p} \epsilon_p(t+1), \tag{14}$$

$$f_p(t+1) = \arg\min_{f_p} L_p^{dual}(t), \tag{15}$$

$$\lambda_p(t+1) = \lambda_p(t) + \frac{\eta}{2} \sum_{j \in \mathcal{N}_p} [f_p(t+1) - f_j(t+1)]. \tag{16}$$

The iterations (14)-(16) are summarized in Algorithm 2, and are illustrated in Figure 1. All nodes have its corresponding value of ρ. Every node $p \in \mathcal{P}$ updates its local estimates $\mu_p(t)$, $f_p(t)$ and $\lambda_p(t)$ at time t; at time $t+1$, node p first perturbs the dual variable $\lambda_p(t)$ obtained at time t to obtain $\mu_p(t+1)$ via (14), and then uses training dataset D_p to compute $f_p(t+1)$ via (15). Next, node p sends $f_p(t+1)$ to all its neighboring nodes. The $(t+1)$-th update is done when each node updates its local $\lambda_p(t+1)$ via (16). We then have the following theorem.

Algorithm 2 Dual Variable Perturbation

```
Required: Randomly initialize $f_p, \lambda_p = \mathbf{0}_{d\times 1}$ for every $p \in \mathcal{P}$
Input: $\hat{D}$, $\{[\alpha_p(1), \alpha_p(2), ...]\}_{p=1}^{P}$
for $t = 0, 1, 2, 3, ...$ do
  for $p = 0$ to $P$ do
    Let $\hat{\alpha}_p = \alpha_p(t) - \ln\left(1 + \frac{c_1}{\frac{B_p}{C^R}\left(\rho+2\eta N_p\right)}\right)^2$.
    if $\hat{\alpha}_p > 0$ then
      $\Phi = 0$.
    else
      $\Phi = \frac{c_1}{\frac{B_p}{C^R}(e^{\alpha_p(t)/4}-1)} - \rho - 2\eta N_p$ and $\hat{\alpha}_p = \alpha_p(t)/2$.
    end if
    Draw noise $\epsilon_p(t)$ according to $\mathcal{K}_p(\epsilon) \sim e^{-\zeta_p(t)\|\epsilon\|}$ with $\zeta_p(t) = \hat{\alpha}_p$.
    Compute $\mu_p(t+1)$ via (14) and $f_p(t+1)$ via (15) with augmented Lagrange function as (13).
  end for
  for $p = 0$ to $P$ do
    Broadcast $f_p(t+1)$ to all neighbors $j \in \mathcal{N}_p$.
  end for
  for $p = 0$ to $P$ do
    Compute $\lambda_p(t+1)$ via (16).
  end for
end for
Output: $\{f_p^*\}_{p=1}^{P}$.
```

Theorem 1. *Under Assumption 1, 2 and 3, if the DR-ERM problem can be solved by Algorithm 2, then Algorithm 2 solving this distributed problem is dynamic α-differential private with $\alpha_p(t)$ for each node $p \in \mathcal{P}$ at time t. Let $Q(f_p(t)|D)$ and $Q(f_p(t)|D'_p)$ be the probability density functions of $f_p(t)$ given dataset D and D'_p, respectively, with $H_d(D, D'_p) = 1$. The ratio of conditional probabilities of $f_p(t)$ is bounded as follows:*

$$\frac{Q(f_p(t)|D)}{Q(f_p(t)|D'_p)} \leqslant e^{\alpha_p(t)}. \tag{17}$$

Proof: *See Appendix.*

4 Performance Analysis

In this section, we discuss the performance of Algorithm 2. We establish performance bounds for regularization functions with l_2 norm. Our analysis is based on the following assumptions:

Assumption 4. The data points $\{(x_{pi}, y_{pi})\}_{i=1}^{B_p}$ are drawn i.i.d. from a fixed but unknown probability distribution $\mathbb{P}^{xy}(x_{pi}, y_{pi})$ at each node $p \in \mathcal{P}$.

Assumption 5. $\epsilon_p(t)$ is drawn from (15) with the same $\alpha_p(t) = \alpha(t)$ for all $p \in \mathcal{P}$ at time $t \in \mathbb{Z}$.

We then define the expected loss of node p using classifier f_p as follows, under Assumption 4:

$$\hat{C}(f_p) := C^R \mathbb{E}_{(x,y)\sim\mathbb{P}^{xy}}(\mathcal{L}(yf^T x)),$$

and the corresponding expected objective function $\hat{Z}$ is:

$$\hat{Z}_p(f_p) := \hat{C}(f_p) + \rho R(f_p).$$

The performance of non-private non-distributed ERM classification learning has been already studied by, for example, Shalev et al. in [18] (also see the work of Chaudhuri et al. in [3]), which introduces a reference classifier f^0 with expected loss $\hat{C}(f^0)$, and shows that if the number of data points is sufficiently large, then the actual expected loss of the trained l_2 regularized support vector machine (SVM) classifier f_{SVM} satisfies

$$\hat{C}(f_{SVM}) \leqslant \hat{C}^0 + \alpha_{acc},$$

where α_{acc} is the generalization error. We use a similar argument to study the accuracy of Algorithm 1. Let f^0 be the reference classifier of Algorithm 1. We quantify the performance of our algorithms with f^* as the final output by the number of data points required to obtain

$$\hat{C}(f^*) \leqslant \hat{C}^0 + \alpha_{acc}.$$

However, instead of focusing on only the final output, we care about the learning performance at all iterations. Let $f_p^{non}(t+1) = \arg\min_{f_p} L_p^N(t)$ be the intermediate updated classifier at t, and let $f^* = \arg\min_{f_p} Z_p(f_p|D_p)$ be the final output of Algorithm 1. From Theorem 9 (see Appendix A), the sequence $\{f_p^{non}(t)\}$ is bounded and converges to the optimal value f^* as time $t \to \infty$. Note that $\{f_p^{non}(t)\}$ is a non-private classifier without added perturbations. Since the optimization is minimization, then there exists a constant $\Delta^{non}(t)$ at time t such that: $\hat{C}(f_p^{non}(t)) - \hat{C}(f^*) \leqslant \Delta^{non}(t)$, and substituting it to $\hat{C}(f^*) \leqslant \hat{C}^0 + \alpha_{acc}$, yields:

$$\hat{C}(f_p^{non}(t)) \leqslant \hat{C}^0 + \Delta^{non}(t) + \alpha_{acc}. \tag{18}$$

Clearly, the above condition depends on the reference classifier f^0; actually, as shown later in this section, the number of data points depends on the l_2-norm $\| f^0 \|$ of the reference classifier. Usually, the reference classifier is chosen with an upper bound on $\| f^0 \|$, say b^0. Based on (18), we provide the following theorem on the performance of Algorithm 1.

Theorem 2. *Let $R(f_p(t)) = \frac{1}{2} \| f_p(t) \|^2$, and let f^0 such that $\hat{C}(f^0) = \hat{C}^0$ for all $p \in \mathcal{P}$ at time t, and $\delta > 0$ is a positive real number. Let $f_p^{non}(t+1) = \arg\min_{f_p} L_p^N(f_p, t|D_p)$ be the output of Algorithm 1. If Assumption 1 and 4 are satisfied, then there exists a constant β_{non} such that if the number of data points, B_p in $D_p = \Big\{(x_{ip}, y_{ip}) \subset \mathbb{R}^d \times \{-1, 1\}\Big\}$ satisfy: $B_p > \beta_{non}\left(\frac{C^R\|f^0\|^2\ln(\frac{1}{\delta})}{\alpha_{acc}^2}\right)$, then $f_p^{non}(t+1)$ satisfies: $\mathbb{P}\big(\hat{C}(f_p^{non}(t+1)) \leqslant \hat{C}^0 + \alpha_{acc} + \Delta^{non}(t)\big) \geqslant 1 - \delta$. for all $t \in \mathbb{Z}_+$.*

Note that $\alpha_{acc} \leqslant 1$ is required for most machine learning algorithms. In the case of SVM, if the constraints are $y_i f^T x_i \leqslant c_{SVM}$, for $i = 1$, ..., n, where n is the number of data points, then, classification margin is $c_{svm}/ \parallel f \parallel$. Thus, if we want to maximization the margin $c_{SVM}/ \parallel f^0 \parallel$ we need to choose a large value of $\parallel f^0 \parallel$. Larger values of $\parallel f^0 \parallel$ are usually chosen for non-separable or with small margin. In the following section, we provide the performance guarantees of Algorithm 2.

4.1 Performance of Private Algorithms

Similar to Algorithm 1, we solve an optimization problem minimizing $L_p^{dual}(f_p, t|D_p)$ at each iteration. Let $f_p(t)$ and $\lambda_p(t)$ be the primal and dual variables used in minimizing $L_p^{dual}(f_p, t|D_p)$ at iteration t, respectively. Suppose that starting from iteration t, the noise vector is static with $\epsilon_p(t)$ generated at iteration t. To compare our private classifier at iteration t with a private reference classifier $f^0(t)$, we construct a corresponding algorithm, Alg-2, associated with Algorithm 2. However, starting from iteration $t+1$, the noise vector in Alg-2 $\epsilon_p(t') = \epsilon_p(t)$ for all $t' > t$. In other words, solving Alg-2 is equivalent to solving the optimization problem with the objective function $Z_p^{dual}(f_p, t|D_p, \epsilon^{pi}(t))$, $t \geqslant 0$ defined as follows:

$$Z_p^{dual}(f_p, t|D_p, \epsilon_p(t)) := Z_p(f_p|D_p) + \frac{C^R}{B_p}\epsilon_p(t) f_p.$$

Let $f_p'(t)$ and $\lambda_p'(t)$ be the updated variables of the ADMM-based algorithm minimizing $Z_p^{dual}(f_p, t|D_p)$ at iteration t. Then, Alg-2 can be interpreted as minimizing

$$Z_p^{dual}(f_p, t|D_p, \epsilon^{pi}(t))$$

with initial condition as $f_p'(0) = f_p(t)$ and $\lambda_p'(0) = \lambda_p(t)$ for all $p \in \mathcal{P}$. Let $Z_p^{dual}(f_p, t|D_p, \epsilon^{pi}(t))$ be regarded as the associated objective function of Alg-2.

Since $Z_p^{dual}(f_p, t|D_p, \epsilon^{pi}(t))$ is real and convex, then, similar to Algorithm 1, the sequence $\{f_p(t)\}$ is bounded and $f_p(t)$ converges to $f_p^*(t)$, which is a limit point of $f_p(t)$. Thus, there exists a constant $\Delta_p^{dual}(t)$ given noise vector $\epsilon_p(t)$ such that

$$\hat{C}(f_p(t)) - \hat{C}(f_p^*(t)) \leqslant \Delta_p^{dual}(t).$$

The performance analysis in Theorem 2 can also used in DVP. Specifically, the performance is measured by the number of data points, B_p, for all $p \in \mathcal{P}$ required to obtain

$$\hat{C}(f_p(t)) \leqslant \hat{C}^0(t) + \alpha_{acc} + \Delta_p^{dual}(t).$$

We say that every learned $f_p(t)$ is α_{acc}-optimal if it satisfies the above inequality.

We now establish the performance bounds for Algorithm 2, DVP, which is summarized in the following theorem.

Theorem 3. *Let $R(f_p(t)) = \frac{1}{2} \parallel f_p(t) \parallel^2$, and $f_p^0(t)$ such that $\hat{C}(f_p^0(t)) = \hat{C}^0(t)$ for all $p \in \mathcal{P}$, and a real number $\delta > 0$. If Assumption 1, 4 and 5 are satisfied, then there exists a constant β_{dual} such that if the number of data points, B_p in $D_p = \Big\{(x_{ip}, y_{ip}) \subset \mathbb{R}^d \times \{-1, 1\}\Big\}$ satisfy:*

$$B_p > \beta_{dual} \max\Bigg(\max_t \Big(\frac{\parallel f_p^0(t+1) \parallel d \ln(\frac{d}{\delta})}{\alpha_{acc}\alpha_p(t)}\Big),$$
$$\max_t \Big(\frac{C^R c_1 \parallel f_p^0(t+1) \parallel^2}{\alpha_{acc}\alpha_p(t)}\Big), \max_t \Big(\frac{C^R \parallel f_p^0(t+1) \parallel^2 \ln(\frac{1}{\delta})}{\alpha_{acc}^2}\Big)\Bigg),$$

then $f_p^*(t+1)$ satisfies:

$$\mathbb{P}\big(\hat{C}(f_p^*(t+1)) \leqslant \hat{C}^0(t+1) + \alpha_{acc}\big) \geqslant 1 - 2\delta.$$

Corollary 3.1. *Let $f_p(t+1) = \arg\min_{f_p} L_p^{dual}(f_p, t|D_p)$ be the updated classifier of Algorithm 2 and let $f_p^0(t)$ be a reference classifier such that $\hat{C}(f_p^0(t) = \hat{C}^0(t)$. If all the conditions of Theorem 3 are satisfied, then $f_p(t+1)$ satisfies*

$$\mathbb{P}\big(\hat{C}(f_p(t+1)) \leqslant \hat{C}^0(t) + \alpha_{acc} + \Delta_p^{dual}(t)\big) \geqslant 1 - 2\delta. \quad (19)$$

Proof. The following inequality holds for $f_p(t)$ and $f_p^*(t)$

$$\hat{C}(f_p(t)) - \hat{C}(f_p^*(t)) \leqslant \Delta_p^{dual}(t),$$

and from Theorem 3,

$$\mathbb{P}\big(\hat{C}(f_p^*(t+1)) \leqslant \hat{C}^0(t+1) + \alpha_{acc}\big) \geqslant 1 - 2\delta.$$

Therefore, we can arrive at (19). □

Theorem 3 and Corollary 3.1 can guarantee the privacy defined in both Definition 1. From Theorem 3, we can see that, for non-separable problems or ones with a small margin, in which a larger $\parallel f_p^0(t) \parallel$ is used, the terms $\frac{1}{\alpha_{acc}}$ and $\parallel f_p^0(t) \parallel$ have a significant influence on the requirement of datasets size for DVP. Moreover, the privacy increases by trading the accuracy. Therefore, It is essential to manage the tradeoff between the privacy and the accuracy, and this will be discussed in Section 5.

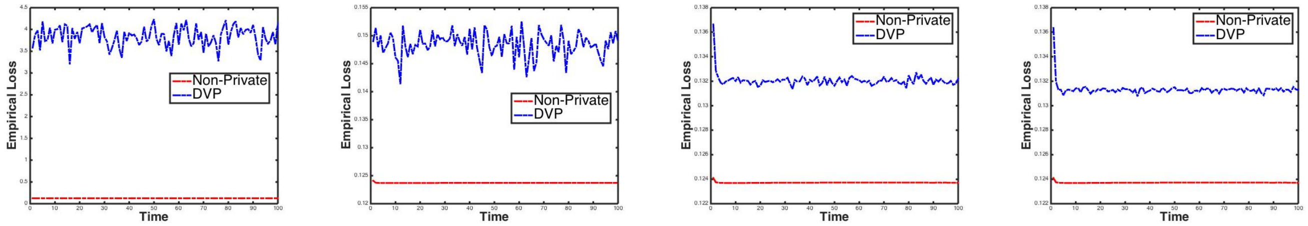
5 Numerical Experiment

In this section, we test Algorithm 2 with real world training dataset. The dataset used is the *Adult* dataset from UCI Machine Learning Repository [1], which contains demographic information such as age, sex, education, occupation, marital status, and native country. In the experiments, we use our algorithm to develop a dynamic differential private logistic regression. The logistic regression, i.e., $\mathcal{L}_{LR}$ takes the following form: $\mathcal{L}_{LR}(y_{ip} f^T x_{ip}) = log(1 + exp(-y_{ip} f_p^T x_{ip}))$, whose first-order derivative and the second-order derivative can be bounded as $|\mathcal{L}_{LR}'| \leqslant 1$ and $|\mathcal{L}_{LR}''| \leqslant \frac{1}{4}$, respectively, satisfying Assumption 3. Therefore, the loss function of logistic regression satisfies the conditions shown in Assumption 2 and 3. In this experiment, we set $R(f_p) = \frac{1}{2} \parallel f_p \parallel^2$, and $c_1 = \frac{1}{4}$. We can directly apply the loss function $\mathcal{L}_{LR}$ to Theorem 1 and 2 with $R(f) = \frac{1}{2} \parallel f_p \parallel^2$, and $c_1 = \frac{1}{4}$, and then it can provide $\alpha_p(t)$-differential privacy for all $t \in \mathbb{Z}$.

We also study the privacy-accuracy tradeoff of Algorithm 2. The privacy is quantified by the value of $\alpha_p(t)$. A larger $\alpha_p(t)$ implies that the ratio of the densities of the classifier $f_p(t)$ on two different data sets is larger, which implies a higher belief of the adversary when one data point in dataset D is changed; thus, it provides lower privacy. However, the accuracy of the algorithm increases as $\alpha_p(t)$ becomes larger. As shown in Figure 2, a larger $\alpha_p(t)$ leads to faster convergence of the algorithm. When $\alpha_p(t)$ is small, the model is more private but less accurate. Therefore, the utilities of privacy and accuracy need to satisfy the following assumptions:

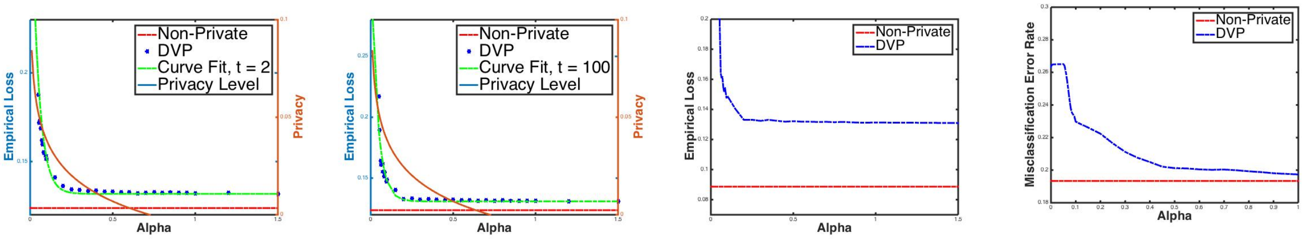
Assumption 6. The utilities of privacy is monotonically increasing with respect to $\alpha_p(t)$ for every $p \in \mathcal{P}$ but accuracy is monotonically decreasing with respect to $\alpha_p(t)$ for every $p \in \mathcal{P}$.

The quality of classifier is measured by the total empirical loss $\overline{C}(t) = \frac{C^R}{B_p}\sum_{i=1}^{B_p} \mathcal{L}(y_{ip} f_p(t)^T x_{ip})$. Let $L_{acc}(\cdot) : \mathbb{R}_+ \to \mathbb{R}$ represent the relationship between $\alpha_p(t)$ and $\overline{C}(t)$. The function L_{acc} is obtained by curve fitting given the experimental data points $(\alpha_p(t), \overline{C}(t))$. Let $U_{priv}(\alpha_p(t)) : \mathbb{R}_+ \to \mathbb{R}$ be the utility of privacy, same for every node $p \in \mathcal{P}$. Besides the decreasing monotonicity, $U_{priv}(\alpha_p(t))$ is assumed to be convex and doubly differentiable function of $\alpha_p(t)$. In our experiment, we model the utility of privacy as: $U_{priv}(\alpha_p(t)) = \omega_{p1} \cdot \ln \frac{\omega_{p2}}{\omega_{p3}\alpha_p(t) + \omega_{p4}\alpha_p^2(t)}$, where, $\omega_{pj} \in \mathbb{R}$ for $j = 1, 2, 3, 4$. For training the classifier, we use a few fixed values of ρ and test the empirical loss $\overline{C}(t) = \frac{C^R}{B_p}\sum_{i=1}^{B_p} \mathcal{L}_{LR}(t)$ of the classifier. Then, we select the value of ρ that minimizes the empirical loss for a fixed α_p (0.3 in this experiment). We also test the non-private version of algorithm, and the corresponding minimum ρ is obtained as the control. We choose the corresponding optimal values of the regularization parameter ρ



(a) $\alpha_p(t) = 0.01$ (b) $\alpha_p(t) = 0.1$ (c) $\alpha_p(t) = 0.5$ (d) $\alpha_p(t) = 1$

Figure 2. Convergence of DVP algorithm, at iteration $t = 100$ (before the stop time) with different values of $\alpha_p(t)$ with $\rho = 10^{-2.5}$ and $C^R = 1750$



(a) DVP: $t = 2$ (b) DVP: $t = 100$ (c) Empirical risk vs. α_p. (d) Misclassification error rate vs. α_p.

Figure 3. Privacy-accuracy tradeoff.(a)-(b): DVP, with $\omega_{p1} = 0.02$, $\omega_{p2} = 6$, $\omega_{p3} = 9$, $\omega_{p4} = 1$ (before the stop time); (c): Privacy-accuracy tradeoff; empirical risk vs. α_p of final optimum output. (d): Privacy-accuracy tradeoff; misclassifications error rate vs. α_p of iteration 100.

as 10^{-10}, $10^{-2.5}$ for Algorithm 1 and 2, respectively. The values of C^R are chosen as 1750 and 1750 for Algorithm 1 and 2, respectively. Figure 2 shows the convergence of DVP at different values of $\alpha_p(t)$ at a given iteration t. Larger values of α_p yield better convergence. However, a larger α_p leads to poorer privacy. Figure 3 (a)-(b) shows the privacy-accuracy tradeoff of DVP at different iterations. By curve fitting, we model the function

$$L_{acc}(\alpha_p(t)) = c_4 \cdot e^{-c_5\alpha_p(t)} + c_6,$$

where c_4, $c_5, c_6 \in \mathbb{R}_+$. From the experimental results, we determine $c_4 = 0.2$, $c_5 = 25$, $c_6 = \min_t\{\overline{C}(t)\}$; these values are applicable at all iterations. For iteration $t > 1$, $c_4 = 20$, $c_5 = 20$, $c_6 = \frac{1}{81}\sum_{t=20}^{100}\overline{C}(t)$. Figure 3 (c)-(d) shows the privacy-accuracy tradeoff of the final optimum classifier in terms of the empirical loss and misclassification error rate (MER). The MER is determined by the fraction of times the trained classifier predicts a wrong label.

6 Conclusion

In this work, we have developed an ADMM-based algorithm, using dual variable perturbation (DVP) to solve a centralized regularized ERM in a distributed fashion while providing dynamic $\alpha(t)$-differential privacy for the ADMM iterations as well as the final trained output. Thus, the sensitive information stored in the training dataset at each node is protected against both the internal and the external adversaries.

Based on distributed training datasets, Algorithm 2 perturbs the dual variable $\lambda_p(t)$ for every node $p \in \mathcal{P}$ at iteration t before minimizing the augmented Lagrange function to calculate the primal variable $f_p(t)$. The performance analysis of DVP indicates that it is suitable for difficult problems that are non-separable or with small margin. In general, the accuracy decreases as privacy requirements are more stringent. The tradeoff between the privacy and accuracy is studied. Our experiments on real data from UCI Machine Learning Repository show that dual variable perturbation is able to manage the privacy-accuracy tradeoff while keeping a good convergence performance.

APPENDIX

Proof. (**Theorem 1**) Let $f_p(t+1)$ be the optimal primal variable with zero duality gap. From the Assumption 1 and 2, we know that both the loss function $\mathcal{L}$ and the regularizer $R(\cdot)$ are differentiable and convex, and by using the Karush-Kuhn-Tucker (KKT) optimality condition, we have

$$\begin{aligned}0 =& \frac{C^R}{B_p}\sum_{i=1}^{B_p} y_{ip}\mathcal{L}'(y_{ip}f_p(t+1)^T x_{ip})x_{ip} + \rho\nabla R(f_p)\\ &+ 2\big(\frac{C^R}{2B_p}\epsilon_p(t) + \lambda_p(t)\big) + (\Phi + 2\eta N_p)f_p(t+1)\\ &- \eta\sum_{i\in\mathcal{N}_p}(f_p(t) + f_i(t)),\end{aligned}$$

from which we can establish the relationship between the noise $\epsilon_p(t)$ and the optimal primal variable $f_p(t+1)$ as:

$$\begin{aligned}\epsilon_p(t) =& -\sum_{i=1}^{B_p} y_{ip}\mathcal{L}'(y_{ip}f_p(t+1)^T x_{ip})x_{ip} - \frac{B_p}{C^R}\rho\nabla R(f_p)\\ &- \frac{2B_p}{C^R}\lambda_p(t) - \frac{B_p}{C^R}(\Phi + 2\eta N_p)f_p(t+1)\\ &+ \frac{B_p\eta}{C^R}\sum_{i\in\mathcal{N}_p}(f_p(t) + f_i(t)).\end{aligned} \tag{20}$$

From the convexity (Assumption 1) of $L_p^{dual}(t)$ is strictly convex, there is a unique value of $f_p(t+1)$ for fixed $\epsilon_p(t)$ and dataset D_p. The equation (20) shows that for any value of $f_p(t+1)$, we can find a unique value of $\epsilon_p(t)$ such that $f_p(t{+}1)$ is the minimizer of L_p^{dual}. Therefore, given a dataset D_p, the relation between $\epsilon_p(t)$ and $f_p(t+1)$ is bijective.

Let D_p and D'_p be two datasets with

$$H_d(D_p, D'_p) = 1, (x_i, y_i) \in D_p$$

and $(x'_i, y'_i) \in D'_p$ are the corresponding two different data points. Let two matrices $\mathbf{J}_f(\epsilon_p(t)|D_p)$ and $\mathbf{J}_f(\epsilon'_p(t)|D'_p)$ denote the Jacobian matrices of mapping from $f_p(t{+}1)$ to $\epsilon_p(t)$ and $\epsilon'_p(t)$, respectively. Then, transformation from noise $f_p(t+1)$ to $\epsilon_p(t)$ by Jacobian yields:

$$\frac{Q(f_p(t+1)|D_p)}{Q(f_p(t+1)|D'_p)} = \frac{q(\epsilon_p(t)|D_p)}{q(\epsilon'_p(t)|D'_p)} \frac{|\det(\mathbf{J}_f(\epsilon_p(t)|D_p))|^{-1}}{|\det(\mathbf{J}_f(\epsilon'_p(t)|D'_p))|^{-1}},$$

where $q(\epsilon_p(t)|D_p)$ and $q(\epsilon'_p(t)|D'_p)$ are the densities of $\epsilon_p(t)$ and $\epsilon'_p(t)$, respectively, given $f_p(t{+}1)$ when the datasets are D_p and D'_p, respectively.

From equation (20), the Jacobian matrix is found as:

$$\begin{aligned}\mathbf{J}_f(\epsilon_p(t)|D_p) = &-\sum_{i=1}^{B_p} \mathbf{J}_f^0(x_i, y_i) - \frac{B_p}{C^R}\rho\nabla^2 R(f_p(t+1)) \\ &- \frac{B_p}{C^R}(\Phi + 2\eta N_p)\mathbf{I}_d.\end{aligned}$$

Let $\mathbf{M} = \mathbf{J}_f^0(x'_i, y'_i) - \mathbf{J}_f^0(x_i, y_i)$, and $\mathbf{H} = -\mathbf{J}_f(\epsilon_p(t)|D_p)$, and thus $\mathbf{J}_f(\epsilon_p(t)|D'_p) = -(\mathbf{M}{+}\mathbf{H})$. Let $h_j(\mathbf{W})$ be the j-th largest eigenvalue of a symmetric matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ with rank θ. Then, we have the fact:

$$\det(\mathbf{I} + \mathbf{W}) = \prod_{j}^{\theta}(1 + h_j(\mathbf{W})).$$

Since the matrix $x_i x_i^T$ has rank 1, matrix $\mathbf{M}$ has rank at most 2; thus matrix $\mathbf{H}^{-1}\mathbf{M}$ has rank at most 2; therefore, we have:

$$\begin{aligned}\det(\mathbf{H} + \mathbf{M}) &= \det(\mathbf{H}) \cdot \det(\mathbf{I} + \mathbf{H}^{-1}\mathbf{M}) \\ &= \det(\mathbf{H}) \cdot (1 + h_1(\mathbf{H}^{-1}\mathbf{M}))(1 + h_2(\mathbf{H}^{-1}\mathbf{M}).\end{aligned}$$

Thus, the ratio of determinants of the Jacobian matrices can be expressed as:

$$\begin{aligned}\frac{|\det(\mathbf{J}_f(\epsilon_p(t)|D_p))|^{-1}}{|\det(\mathbf{J}_f(\epsilon'_p(t)|D'_p))|^{-1}} =& \frac{|\det(\mathbf{H} + \mathbf{M})|}{|\det(\mathbf{H})|} \\ =&|\det(\mathbf{I} + \mathbf{H}^{-1}\mathbf{M})| \\ =&(1 + h_1(\mathbf{H}^{-1}\mathbf{M}))(1 + h_2(\mathbf{H}^{-1}\mathbf{M}) \\ =&|1 + h_1(\mathbf{H}^{-1}\mathbf{M}) + h_2(\mathbf{H}^{-1}\mathbf{M}) \\ &+ h_1(\mathbf{H}^{-1}\mathbf{M})h_2(\mathbf{H}^{-1}\mathbf{M})|.\end{aligned}$$

Based on Assumption 2, all the eigenvalues of $\nabla^2 R(f_p(t+1))$ is greater than 1 [16]. Thus, from Assumption 1, matrix $\mathbf{H}$ has all eigenvalues at least $\frac{B_p}{C^R}\left(\rho + \Phi + 2\eta N_p\right)$. Therefore, $|h_1(\mathbf{H}^{-1}\mathbf{M})| \leqslant \frac{|h_i(\mathbf{M})|}{\frac{B_p}{C^R}\left(\rho+\Phi+2\eta N_p\right)}$.

Let $\sigma_i(\mathbf{M})$ be the non-negative singular value of the symmetric matrix $\mathbf{M}$. According to [4], we have the inequality $\sum_i |h_i(\mathbf{M})| \leqslant \sum_i \sigma_i(\mathbf{M})$. Thus, we have $|h_1(\mathbf{M})|+|h_2(\mathbf{M})| \leqslant \sigma_1(\mathbf{M}) + \sigma_2(\mathbf{M})$. Let $\parallel X \parallel_\Sigma = \sum_i \sigma_i$ be the trace norm of X. Then, according to the *trace norm inequality*, we have:

$$\parallel \mathbf{M} \parallel_\Sigma \leqslant \parallel \mathbf{J}^0(x'_i, y'_i) \parallel_\Sigma + \parallel -\mathbf{J}^0(x_i, y_i) \parallel_\Sigma .$$

As a result, based on the upper bounds from Assumption 1 and 3, we have:

$$\begin{aligned}|h_1(\mathbf{M})| + |h_2(\mathbf{M})| &\leqslant \parallel \mathbf{J}^0(x'_i, y'_i) \parallel_\Sigma + \parallel -\mathbf{J}^0(x_i, y_i) \parallel_\Sigma \\ &\leqslant |(y_i^2 \mathcal{L}''(y_i f_p(t+1)^T x_i)| \cdot \parallel x_i \parallel \\ &+ |(y_i'^2 \mathcal{L}''(y'_i f_p(t+1)^T x'_i)| \cdot \parallel x'_i \parallel \\ &\leqslant 2c_1,\end{aligned}$$

which follows $h_1(\mathbf{M})h_2(\mathbf{M}) \leqslant c_1^2$. Finally, the ratio of determinants of Jacobian matrices is bounded as:

$$\begin{aligned}\frac{|\det(\mathbf{J}_f(\epsilon_p(t)|D_p))|^{-1}}{|\det(\mathbf{J}_f(\epsilon'_p(t)|D'_p))|^{-1}} &\leqslant (1 + \frac{c_1}{\frac{B_p}{C^R}\left(\rho + \Phi + 2\eta N_p\right)})^2 \\ &= e^{\overline{\alpha}},\end{aligned} \tag{21}$$

where $\overline{\alpha} = \ln\left(1 + \frac{c_1}{\frac{B_p}{C^R}\left(\rho+\Phi+2\eta N_p\right)}\right)^2$.

Now, we bound the ratio of densities of $\epsilon_p(t)$. Let $sur(E)$ be the surface area of the sphere in d dimension with radius E, and $sur(E) = sur(1) \cdot E^{d-1}$. We can write

$$\frac{q(\epsilon_p(t)|D_p)}{q(\epsilon'_p|D'_p)} = \frac{\mathcal{K}(\epsilon_p(t))\frac{\|\epsilon_p(t)\|^{d-1}}{sur(\|\epsilon_1(t)\|)}}{\mathcal{K}(\epsilon'_p(t))\frac{\|\epsilon'_p(t)\|^{d-1}}{sur(\|\epsilon'_p(t)\|)}} \leqslant e^{\zeta_p(t)(\|\epsilon'_p(t)\|-\|\epsilon_p(t)\|)} \leqslant e^{\hat{\alpha}_p}, \tag{22}$$

where $\hat{\alpha}_p$ is a constant satisfying the above inequality. For non-negative Φ, let $\hat{\alpha}_p = \alpha_p(t) - \ln\left(1 + \frac{c_1}{\frac{B_p}{C^R}\left(\rho+2\eta N_p\right)}\right)^2$. If $\hat{\alpha}_p > 0$, then we fix $\Phi = 0$, and thus $\hat{\alpha}_p = \alpha_p(t) - \overline{\alpha}$. Otherwise, let $\Phi = \frac{c_1}{\frac{B_p}{C^R}(e^{\alpha_p(t)/4}-1)} - \rho - 2\eta N_p$, and $\hat{\alpha}_p = \frac{\alpha_p(t)}{2}$, then $\hat{\alpha}_p = \alpha_p(t) - \overline{\alpha}$. Therefore, we obtain

$$\frac{|\det(\mathbf{J}_f(\epsilon_p(t)|D_p))|^{-1}}{|\det(\mathbf{J}_f(\epsilon'_p(t)|D'_p))|^{-1}} \leqslant e^{\alpha_p(t)-\hat{\alpha}_p}.$$

From the upper bounds stated in Assumption 1 and 3, the l_2 norm of the difference of ϵ_1 and ϵ_2 can be bounded as:

$$\begin{aligned}\parallel \epsilon'_p(t) - \epsilon_p(t) \parallel = \sum_{i=1}^{B_p} &\parallel y_{ip}\mathcal{L}'(y'_{ip} f_p(t+1)^T x'_{ip})x'_{ip} \\ &- (y_{ip}\mathcal{L}'(y_{ip} f_p(t+1)^T x_{ip})x_{ip} \parallel \leqslant 2.\end{aligned}$$

Thus, $\parallel \epsilon'_p(t) \parallel - \parallel \epsilon_p(t) \parallel \leqslant \parallel \epsilon'_p(t) - \epsilon_p(t) \parallel \leqslant 2$. Therefore, by selecting $\zeta_p(t) = \frac{\hat{\alpha}_p}{2}$, we can bound the ratio of conditional densities of $f_p(t+1)$ as

$$\frac{Q(f_p(t+1)|D_p)}{Q(f_p(t+1)|D'_p)} \leqslant e^{\alpha_p(t)},$$

and prove that the DVP can provide $\alpha_p(t)$-differential privacy. □

Acknowledgement

We would like to thank National Science Foundation (CNS-1544782) and NYU University Research Challenge Fund (URCF) that partially support this research.

A References

[1] A. Asuncion and D. Newman. Uci machine learning repository, 2007.

[2] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the sulq framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138. ACM, 2005.

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

[4] P. Chilstrom. Singular value inequalities: New approaches to conjectures. 2013.

[5] M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8. Association for Computational Linguistics, 2002.

[6] Y.-H. Dai. A perfect example for the bfgs method. *Mathematical Programming*, 138(1-2):501–530, 2013.

[7] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

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

[9] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. *Information Systems*, 29(4):343–364, 2004.

[10] P. A. Forero, A. Cano, and G. B. Giannakis. Consensus-based distributed support vector machines. *The Journal of Machine Learning Research*, 11:1663–1707, 2010.

[11] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[12] J. Kim and W. Winkler. Multiplicative noise for masking continuous data. *Statistics*, page 01, 2003.

[13] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su. Scaling distributed machine learning with the parameter server. In *Proc. OSDI*, pages 583–598, 2014.

[14] R. McDonald, K. Hall, and G. Mann. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464. Association for Computational Linguistics, 2010.

[15] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 94–103. IEEE, 2007.

[16] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125. IEEE, 2008.

[17] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84. ACM, 2007.

[18] S. Shalev-Shwartz and N. Srebro. Svm optimization: inverse dependence on training set size. In *Proceedings of the 25th international conference on Machine learning*, pages 928–935. ACM, 2008.

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