Differentially Private Label Protection in Split Learning

Xin Yang∗
yangxin.yx@bytedance.com

Jiankai Sun
jiankai.sun@bytedance.com

Yuanshun Yao
kevin.yao@bytedance.com

Junyuan Xie
junyuan.xie@bytedance.com

Chong Wang
chong.wang@bytedance.com

March 7, 2022

Abstract

Split learning is a distributed training framework that allows multiple parties to jointly train a machine learning model over vertically partitioned data (partitioned by attributes). The idea is that only intermediate computation results, rather than private features and labels, are shared between parties so that raw training data remains private. Nevertheless, recent works showed that the plaintext implementation of split learning suffers from severe privacy risks that a semi-honest adversary can easily reconstruct labels. In this work, we propose TPSL (Transcript Private Split Learning), a generic gradient perturbation based split learning framework that provides provable differential privacy guarantee. Differential privacy is enforced on not only the model weights, but also the communicated messages in the distributed computation setting. Our experiments on large-scale real-world datasets demonstrate the robustness and effectiveness of TPSL against label leakage attacks. We also find that TPSL have a better utility-privacy trade-off than baselines.

1 Introduction

In the past decade, deep learning has achieved tremendous success in many applications including recommendation systems, autonomous driving and healthcare. Nevertheless, the abundant data needed to train deep learning models draws increasing concerns over user privacy, and strict regulations like CCPA1, HIPAA2 and GDPR3 have been introduced to regulate how data can be transmitted and used. To address privacy concerns, Federated Learning (FL) [McMahan et al., 2017, Hanzely et al., 2020, Yuan and Ma, 2020, Ghosh et al., 2020] is proposed so that multiple entities can collaborate to train a model without sharing raw data.

One important category of FL is vertical Federated Learning or vFL [Yang et al., 2019b], where each party has training data of different subjects, but the attributes (e.g. features and labels) of the data are the same. vFL arises in many scenarios like finance and healthcare [Yang et al., 2019b, Li et al., 2022]. Some vFL frameworks apply secure multi-party computation (MPC) protocols including oblivious transfer, garbled circuits [Gascón et al., 2016] and Homomorphic Encryption

∗Correspondence to yangxin.yx@bytedance.com
1California Consumer Privacy Act
2Health Insurance Portability and Accountability Act
3General Data Protection Regulation, European Union
(HE) [Hardy et al., 2017, Yang et al., 2019a,b] to protect data privacy. However, this will cause significant overhead in computation and communication [Kim et al., 2018]. Hence such methods only works with simple models like linear regression and logistic regression.

For larger neural network models, Gupta and Raskar [2018] proposed the idea of split neural network, or split learning. This is also the focus of this paper. The goal of split learning is to train a neural network with the standard centralized training algorithms like mini-batch stochastic gradient descent (SGD) in an MPC setting, while no raw data is shared between parties. To do so, a neural network is split into multiple sub-networks, and each party holds one such sub-network. All the parties simulate centralized training as follows: each party collects input data from either other pieces or raw input owned by the party itself, then they do the local computation within the piece, and finally they send the output (e.g. embeddings or gradients) to other parties that need those intermediate results. Compared to secure MPC methods, split learning is straightforward to implement, and it has small computational/communication cost that is similar to centralized training.

The security of split learning is based on the assumption that neural networks behave like black-boxes and intermediate computation results are so complicated to parse that attackers cannot use them to recover the raw training data. Nevertheless, recently Li et al. [2022] showed that the adversarial party can recover private labels with high accuracy by computing the norm and direction of the gradients. This is not totally unexpected. After all, there is no provable privacy guarantee for split learning. In order to apply split learning in practice, we need to build the learning algorithms on theoretical foundations.

Differential privacy (DP) [Dwork et al., 2006, Dwork and Roth, 2014] is a widely applied measurement of privacy and has been deployed in many real-world applications [Ding et al., 2017, Abowd, 2018]. DP has become the standard practice in privacy-preserving machine learning [Chaudhuri et al., 2011, Lei, 2011, Abadi et al., 2016]. It is natural to ask if we can bring DP into the split learning setting. However, one major difference of split learning (or the more general vFL setting) between traditional machine learning algorithms is that the parties in split learning need to exchange messages (transcripts) and hence more information is exposed besides the model parameters. This is a new challenge for designing privacy-preserving learning protocols.

In this work, we present Transcript Private Split Learning, or TPSL, a novel framework of split learning with strong provable DP guarantee. In particular, we focus on protecting label information with DP. This is because not only label leakage is a known privacy threat in split learning [Li et al., 2022], but also there are many cases where labels are sensitive and require protection, like online advertising and recommendation systems [Ghazi et al., 2021].

The security of TPSL is provided by our rigorous and complete proof on DP guarantee. The computation and communication cost of TPSL is similar to the naive implementation of split learning, as differential privacy can be efficiently enforced by noise perturbation mechanisms. Inspired by the white-box attack of split learning [Erdogan et al., 2021], TPSL adds suitable noise to the gradients so that the given privacy budget is met. Compared to the baseline of adding isotropic noise on the transcripts, TPSL deals with the perturbation more carefully and efficiently. Experiments over large-scale data verify the effectiveness of TPSL to achieve better utility-privacy trade-off.

2 Background

2.1 Split Learning

Split learning [Vepakomma et al., 2018, Gupta and Raskar, 2018, Abuadbba et al., 2020, Ceballos et al., 2020] is a distributed deep learning framework that allows multiple parties to jointly train a
neural network without directly sharing raw data. In this work, we focus on two-party split learning over binary classification problem, where the domain of features is \( \mathcal{X} \) and the label is in \( \mathcal{Y} = \{0,1\} \). For simplicity of exposition we consider the scenario where the training data is perfectly vertically split. Following the notations in Li et al. [2022], \( P_L \) denotes the label party that holds all the labels and \( P_N \) denotes the non-label party that holds all the features. We assume that at the beginning of split learning, features and labels have already been aligned by protocols such as private set intersection [Kolesnikov et al., 2016, Pinkas et al., 2018].

The goal of the two parties is to jointly train a prediction model \( h_{\theta_L} \circ f_{\theta_N} \). Here \( f : \mathcal{X} \to \mathbb{R}^d \) is the feature representation function held by \( P_N \), \( \theta_N \) is the model parameters of \( P_N \) and \( d \) is the dimension of the intermediate representation (embeddings). Similarly, \( h : \mathbb{R}^d \to [0,1] \) is the prediction function held by \( P_L \) and \( \theta_L \) is the model parameters of \( P_L \).

In this paper, we consider \( h, f \) are neural networks with nonlinear activation functions, such as ReLU. We consider the standard mini-batch stochastic gradient descent (SGD) training with loss function \( \ell \). For the ease of exposition, we consider an online fashion of training in the sense that each training sample is used once, which is presented in Algorithm 1. On a high level, in the forward phase, \( P_N \) computes the embeddings \( f(x) \) and passes them to \( P_L \); \( P_L \) continues to compute the prediction and loss. In the backward phase, \( P_L \) computes the gradients \( g = \frac{\partial \ell(h(f(x), y))}{\partial f(x)} \) and return them to \( P_N \). Finally, \( P_L \) and \( P_N \) update their models with the gradients.

| Algorithm 1: Stochastic Gradient Descent in Split Learning |
|----------------------------------------------------------|
| **Input:** Features \( X \) held by \( P_N \). Labels \( Y \) held by \( P_L \). \( |X| = |Y| = n \). |
| **Parameters:** Number of batches \( B \). Step size \( \eta \). Loss function \( \ell : [0,1] \times \{0,1\} \to \mathbb{R}_{\geq 0} \). |
| 1 Training samples are divided into \( B \) batches. |
| 2 \( \theta_N, \theta_L \) are initialized with Gaussian initialization. |
| 3 for \( b = 1,2,\cdots,B \) do |
| // Forward phase |
| 4 \( P_N \) computes embeddings \( e_{b,i} := f_{\theta_N}(x_{b,i}) \) for \( i = 1,\cdots,n/B \) where \( x_{b,1}, x_{b,2},\cdots,x_{b,n/B} \) are the features of samples in batch \( b \). |
| 5 \( P_N \) sends \( \{e_{b,1},\cdots,e_{b,n/B}\} \) to \( P_L \). |
| // Backward phase |
| 6 \( P_L \) computes the prediction of the neural network \( h \) as \( \tilde{y}_{b,i} := h_{\theta_L}(f_{\theta_N}(x_{b,N})) \) for \( i = 1,\cdots,n/B \). |
| 7 \( P_L \) computes losses \( \ell_{b,i} := \ell(\tilde{y}_{b,i}, y_{b,i}) \) for \( i = 1,\cdots,n/B \) where \( y_{b,1}, y_{b,2},\cdots,y_{b,n/B} \) are the labels in batch \( b \). |
| 8 \( P_L \) computes gradients \( g_{b,i} := \frac{\partial \ell_{b,i}^{L}}{\partial \theta_{b,i}} \) for \( i = 1,\cdots,n/B \). |
| 9 \( P_L \) sends \( \{g_{b,1},\cdots,g_{b,n/B}\} \) to \( P_N \). |
| // Model updates |
| 10 \( P_L \) updates \( \theta_L \) by \( \theta_L \leftarrow \theta_L - \eta \cdot \frac{B}{n} \cdot \sum_{i=1}^{n/B} \frac{\partial \ell_{b,i}^{L}}{\partial \theta_{b,i}} \). |
| 11 \( P_N \) computes \( \frac{\partial \ell_{b,i}^{L}}{\partial \theta_N} = g_{i}^\top \frac{\partial \ell_{b,i}^{L}}{\partial \theta_{L}} \) and then updates \( \theta_N \) by \( \theta_N \leftarrow \theta_N - \eta \cdot \frac{B}{n} \cdot \sum_{i=1}^{n/B} \frac{\partial \ell_{b,i}^{L}}{\partial \theta_{L}} \). |

### 2.2 Differential Privacy

Differential privacy (DP) [Dwork et al., 2006, Dwork and Roth, 2014] is a quantifiable and rigorous privacy framework. We adopt the following definition of DP.

**Definition 2.1** (Differential Privacy). Fix \( \epsilon, \delta \in \mathbb{R}_{\geq 0} \). We say that a randomized mechanism \( \mathcal{M} \) is
$(\epsilon, \delta)$-differentially private, if for any neighboring datasets $D, D'$, and for any subset $S$ of possible output of $M$,

$$\Pr[M(D) \in S] \leq e^\epsilon \cdot \Pr[M(D') \in S] + \delta.$$  

In split learning, which is a type of MPC, we would like to protect privacy of information shared between the parties. For an MPC protocol $A$, we denote $\text{tran}(A)$ as the transcript of $A$, namely the messages shared between the parties. In split learning, the transcript includes (1) embeddings in the forward phase $f(x)$ and (2) gradients in the backward phase $g = \frac{\partial h(f(x), y)}{\partial f(x)}$. We can use DP to measure the privacy of information exchange in MPC protocols, which we call transcript differential privacy:

**Definition 2.2 (Transcript Differential Privacy).** Fix $\epsilon, \delta \in \mathbb{R}_{\geq 0}$. We say that a randomized mechanism $A$ between multiple parties is $(\epsilon, \delta)$-transcript differentially private, if $\text{tran}(A)$ is $(\epsilon, \delta)$-differentially private.

The notion of neighboring datasets determines the privacy that DP aims to protect. As we focus on label protection, we say two datasets $D, D'$ are neighbouring if they only differ at one label in the samples. This definition of neighbouring dataset is similar to the setting of label DP [Ghazi et al., 2021, Malek Esmaeili et al., 2021].

Sensitivity in differential privacy is defined as the following: Fix $p \geq 0$, the $\ell_p$ sensitivity of function $f$ is $\max_{D, D'} \|f(D) - f(D')\|_p$ where the maximization is over neighbouring dataset $D, D'$.

### 2.3 Threat Model

As the defender, the label party $P_L$ wants to protect the labels from the attacker (non-label party $P_N$). We consider the attacker is semi-honest who faithfully executes the split learning protocol (e.g., no adversarial chosen inputs), but it aims to infer the labels held by $P_L$ from the information received during the protocol execution. In terms of attacker’s capability, we consider two threat models: **black-box** attack and **white-box** attack. In black-box setting, the attacker only has access to its own inputs (features set $X$ and model parameters $\theta_N$) and the transcript $\text{tran}(A)$. As shown by Li et al. [2022], the attacker can still infer the labels based on gradient information under the black-box setting. In white-box setting, the attacker can also access $\theta_L$. In our design, we leverage insights coming from the white-box attack because it is the strongest attack assumption that should be considered in order to design a strong defense.

### 2.4 Notations

For positive integers $m < n$, $[n]$ represents the set $\{1, 2, \cdots, n\}$, and $[m, n]$ represents the set $\{m, m + 1, \cdots, n\}$. Let $\mathcal{N}(\theta, \Sigma)$ denote the (multi-variate) normal distribution with mean $\theta$ and covariance $\Sigma$. For a vector $v$ and a set $S$ that is a subset of the underlying coordinates of $v$, $[v]_S$ represents the restriction of $v$ on $S$, i.e., if $S = \{s_1, s_2, \cdots, s_t\}$, then $[v]_S = (v_{s_1}, v_{s_2}, \cdots, v_{s_t})$.

### 3 Design of Differentially Private Noise

#### 3.1 Gaussian mechanism under white-box attack

Gaussian mechanism [Dwork and Roth, 2014] is a common post-hoc mechanism to implement DP. Let $f$ be a deterministic function with range $\mathbb{R}^m$, and $\Delta f$ be the $\ell_2$ sensitivity of $f$. To achieve DP, Gaussian mechanism adds a Gaussian noise $R \in \mathbb{R}^m$ to the output of $f$. It is shown that
when \( R \sim \mathcal{N}(0, \Delta^2 \sigma^2 \cdot I_m) \) with \( \sigma \geq \frac{\sqrt{2 \ln(1.25/\delta)}}{\epsilon} \), the randomized function \( f^{dp}(D) := f(D) + R \) is \((\epsilon, \delta)\)-DP.

To ensure \( \text{tran}(A) \) to be DP, a natural choice is to apply Gaussian mechanism on \( \text{tran}(A) \), and for the label party \( P_L \) who wants to protect its label, it is reasonable to add noise to the part of the \( \text{tran}(A) \) that \( P_L \) owns, namely the gradients. This approach is also one of our baselines in the experimental section.

To improve from the vanilla Gaussian mechanism, we consider what the adversary might do when facing the basic Gaussian mechanism protection under the strong white-box capability. This helps us design the protection method against that strong white-box attack, which can bring insights on designing transcript DP split learning protocols.

Fix a training sample \((x, y)\), the adversarial \( P_N \) knows the feature \( x \) and its forwarded embedding \( f_{\theta_N}(x) \). It also receives a perturbed gradient \( \tilde{g} := g_y + R \) after \( P_L \)'s DP protection , where \( g_y \in \mathbb{R}^d \) is the correct gradient with respect to the label \( y \) held by \( P_L \), and \( R \sim \mathcal{N}(0, \sigma^2 I_d) \) is the Gaussian noise added with suitable variance \( \sigma^2 \).

Under the white-box assumption, when \( P_N \) knows \( \theta_L \), it can perform the following attack: \( P_N \) can guess the \( y \) that corresponds to the gradient with shorter distance to the perturbed gradient [Erdogan et al., 2021]. Formally, \( P_N \) can first compute the gradients with respect to label \( 0, 1 \) by

\[
\begin{align*}
g_0 & := \frac{\partial \ell(h_{\theta_L}(f_{\theta_N}(x)), 0)}{\partial f_{\theta_N}(x)}, \\
g_1 & := \frac{\partial \ell(h_{\theta_L}(f_{\theta_N}(x)), 1)}{\partial f_{\theta_N}(x)}.
\end{align*}
\]

Then shortest distance attack guesses the value of \( y \) as

\[
\tilde{y} := \begin{cases} 
0 & \text{If } \| \tilde{g} - g_0 \|_2 \leq \| \tilde{g} - g_1 \|_2, \\
1 & \text{Else}.
\end{cases}
\]

We name this attack shortes\( \text{t} \) distance attack. If there is no protection, then shortest distance attack can obtain the correct label \( y \) almost surely. Now we explain how we can leverage shortest distance attack to design an improved mechanism. Notice that the decision of shortest distance attack is merely determined by the difference \( \| \tilde{g} - g_0 \|_2^2 - \| \tilde{g} - g_1 \|_2^2 \). By expanding this difference, we have

\[
\begin{align*}
\| \tilde{g} - g_0 \|_2^2 - \| \tilde{g} - g_1 \|_2^2 &= \langle g_1 - g_0, 2\tilde{g} - g_0 - g_1 \rangle \\
&= \langle g_1 - g_0, 2g_y + R - g_y - g_1 - y \rangle \\
&= \langle g_1 - g_0, g_y - g_1 - y \rangle + 2\langle g_1 - g_0, R \rangle.
\end{align*}
\]

The first term \( \langle g_1 - g_0, g_y - g_1 - y \rangle \) is determined by the true label \( y \) and does not depend on the noise \( R \). For the second term, notice that \( \langle g_1 - g_0, R \rangle \) only depends on the projection of \( R \) in the direction of \( g_1 - g_0 \). In other words, components of \( R \) on other orthogonal directions do not affect the decision of shortest distance attack.

The above calculation suggests that \( g_0 - g_1 \) is a special direction, which inspires us to consider noise perturbation mechanism that focuses on the direction of \( g_0 - g_1 \).

### 3.2 Generic label protection mechanism

Based on the insights inspired by the white-box attack, we propose \textbf{GradPerturb}, a general gradient-perturbation scheme for \( P_L \) in Algorithm 2. \textbf{GradPerturb} is specified by a distribution \( U \) over \( \mathbb{R} \) which
Algorithm 2: GradPerturb equipped with distribution $U$

**Input:** True label $y \in \{0, 1\}$. Gradients $g_0$ and $g_1$ with respect to label 0 and 1.
1 $P_L$ samples $u \sim U$.
2 return $g_y + u \cdot (g_{1-y} - g_y)$.

controls the noise scale. With gradients $g_0$ and $g_1$, GradPerturb adds noise to $g_y$ in the direction of $g_0 - g_1$.

We explore two choices of the distribution $U$:

- (Laplace perturbation) $U$ is $\text{Lap}(b)$, i.e., the Laplace distribution centered at 0 with scale $b$.
  The PDF of $\text{Lap}(b)$ is given by $p(x|b) = \frac{1}{2b}e^{-\frac{|x|}{b}}$.

- (Discrete perturbation) $U$ is a Bernoulli Distribution $\text{Bern}(p)$ for $p \in [0, \frac{1}{2}]$. Namely, $U$ take values from $\{0, 1\}$ with $\Pr[U = 1] = p$ and $\Pr[U = 0] = 1 - p$.

We will show that both mechanisms are differential private. The detailed proof is deferred to Appendix B.

**Lemma 3.1** (Laplace perturbation is differentially private). Let $U = \text{Lap}(b)$. Fix $\epsilon > 0$. When $b \geq 1/\epsilon$, GradPerturb is $(\epsilon, 0)$-DP with respect to the label $y$.

**Lemma 3.2** (Discrete perturbation is differentially private). Let $U = \text{Bern}(p)$. Then GradPerturb is $(\ln \frac{1-p}{p}, 0)$-DP with respect to the label $y$.

**Remark.** It turns out that the discrete perturbation coincides with the randomized response mechanism used in Ghazi et al. [2021]. Indeed, our definition of transcript DP is closely related to the concept of label DP [Ghazi et al., 2021], because we require neighbouring datasets to differ at only one label. Consequently, transcript DP is a stronger notion of privacy in the sense that given any MPC protocol that is $(\epsilon, \delta)$-transcript DP, we can change it into a centralized $(\epsilon, \delta)$-label DP algorithm. It is an interesting question that if the converse is true, namely does there exist a generic reduction that can turn a label DP algorithm into a distributed algorithm with transcript DP guarantee?

## 4 Differentially private split learning

We now use GradPerturb as a building block to design a transcript differentially private split learning algorithm.

### 4.1 Transcript Private Split Learning

Our method TPSL (Transcript Private Split Learning) is presented in Algorithm 3. Compared to the original split learning algorithm (Algorithm 1), TPSL perturbs not only the gradients $g_{b,i}$, but also the model updates of $P_L$. Note that we need to also perturb the model updates so that in our proof we can avoid DP composition theorems [Dwork et al., 2010] that would lead to a DP budget that depends on the number of batches or the number of samples.

The privacy of TPSL is given by the following lemma:

**Lemma 4.1.** Fix $\epsilon > 0$. If GradPerturb is $(\epsilon, 0)$-DP under the distribution $U$, then TPSL is $(2\epsilon, 0)$-transcript DP.
Formally, define \( b \) the same. Formally, let \( (b, i) \) be the index of the different label in \( D \), i.e., \( y_{b^*, i^*} = 1 - y_{b^*, i} \). Then in batch \( 1, 2, \cdots, b^* - 1 \), \( D \) and \( D' \) are identical, hence the probability restricted on the first \( b^* - 1 \) batches is the same. Formally, let \( s_\_ \) be the parts of \( S_t, S_m \) that are in the first \( b^* - 1 \) batches, then we have

\[
\Pr[(S_t, S_m)(D) = s] \leq e^{2c} \cdot \Pr[(S_t, S_m)(D') = s].
\]  

(1)

Let \( (b^*, i^*) \) be the index of the different label in \( D \) and \( D' \), i.e., \( y_{b^*, i^*} = 1 - y_{b^*, i} \). Then in batch 1, 2, \cdots, \( b^* - 1 \), \( D \) and \( D' \) are identical, hence the probability restricted on the first \( b^* - 1 \) batches is the same. Formally, let \( s_\_ \) be the parts of \( S_t, S_m \) that are in the first \( b^* - 1 \) batches, then we have

\[
\Pr[(S_t, S_m)(D) = s]_\_ = \Pr[(S_t, S_m)(D') = s]_\_.
\]

Furthermore, if \( S_t \) and \( S_m \) are the same for the first \( b^* \) batches, then the model weights after the \( b^* \)-th batch will be the same for \( D \) and \( D' \), because all previous model updates are the same. Since the rest training data is identical in \( D \) and \( D' \), the remaining part in \( (S_t, S_m) \) will also be identical. Formally, define \( S_\ast, S_+ \) as

\[
S_\ast = \{e_{b, i}\}_{b \in [b^*+1, B], i \in [1, n/B]} \cup \{\tilde{g}_{b, i}\}_{b \in [1, n/B]} \cup \{\Delta \theta_N^{(b^*)}, \Delta \theta_L^{(b^*)}\},
\]

\[
S_+ = \{e_{b, i}\}_{b \in [b^*+1, B], i \in [1, n/B]} \cup \{\tilde{g}_{b, i}\}_{b \in [b^*+1, B], i \in [1, n/B]}
\]

\[
\cup \{\Delta \theta_N^{(b)}, \Delta \theta_L^{(b)}\}_{b \in [b^*+1, B]}.
\]
Then we have
\[
\Pr \left[ \left( (S_t, S_m)(D) \right) \in S \mid \left( (S_t, S_m)(D) \right) \in S_{-} \cup S_{*} = [s]_{S_{-} \cup S_{*}} \right] = \Pr \left[ \left( (S_t, S_m)(D') \right) \in S \mid \left( (S_t, S_m)(D') \right) \in S_{-} \cup S_{*} = [s]_{S_{-} \cup S_{*}} \right].
\]
So we conclude that
\[
\frac{\Pr \left[ (S_t, S_m)(D) = s \right]}{\Pr \left[ (S_t, S_m)(D') = s \right]} = \frac{\Pr \left[ \left( (S_t, S_m)(D) \right) \in S \mid \left( (S_t, S_m)(D) \right) \in S_{-} \cup S_{*} = [s]_{S_{-}} \right]}{\Pr \left[ \left( (S_t, S_m)(D') \right) \in S \mid \left( (S_t, S_m)(D') \right) \in S_{-} \cup S_{*} = [s]_{S_{-}} \right]}.
\]

The RHS is concerning the ratio of the probability that the transcript and model updates in the \( b^* \)-th batch are the same. Let \( g \) be the value of \( \tilde{g}_{b^*} \) in \( s \), and \( u \) be the value of \( \tilde{u}_{b^*} \) in \( s \). Let \( E_{D,g,u} \) be the event of \( \text{GradPerturb}(y_{b^*}, i^*, g, 0, g_1) = g \), \( \text{GradPerturb}(y_{b^*}, i^*, u_0, u_1) = u \), and \( E_{D',g,u} \) be the event of \( \text{GradPerturb}(1 - y_{b^*}, i^*, g, 0, g_1) = g \), \( \text{GradPerturb}(1 - y_{b^*}, i^*, u_0, u_1) = u \). Since all the training samples except for the \( i^* \)-th one is the same in \( D \) and \( D' \), we have
\[
\frac{\Pr \left[ \left( (S_t, S_m)(D) \right) \in S \mid \left( (S_t, S_m)(D) \right) \in S_{-} \cup S_{*} = [s]_{S_{-}} \right]}{\Pr \left[ \left( (S_t, S_m)(D') \right) \in S \mid \left( (S_t, S_m)(D') \right) \in S_{-} \cup S_{*} = [s]_{S_{-}} \right]} = \frac{\Pr \left[ E_{D,g,u} \right]}{\Pr \left[ E_{D',g,u} \right]}.
\]
Since \( \text{GradPerturb} \) is \((\epsilon, 0)\)-DP with the distribution \( U \), we have
\[
\frac{\Pr \left[ E_{D,g,u} \right]}{\Pr \left[ E_{D',g,u} \right]} \leq e^{2\epsilon},
\]
which completes the proof of Eq. (1).

Then we show \((S_t, S_m)\) is \((2\epsilon, 0)\)-DP. For any subset \( S \) of possible assignments of \((S_t, S_m)\), we have
\[
\Pr \left[ (S_t, S_m)(D) \in S \right] = \int_{s \in S} \Pr \left[ (S_t, S_m)(D) = s \right] ds
\]
\[
\leq \int_{s \in S} e^{2\epsilon} \Pr \left[ (S_t, S_m)(D') = s \right] ds
\]
\[
= e^{2\epsilon} \Pr \left[ (S_t, S_m)(D') \in S \right],
\]
where the second step uses Eq. (1). Hence \((S_t, S_m)\) is \((2\epsilon, 0)\)-DP. As a consequence, TPSL is \((2\epsilon, 0)\)-transcript DP, which completes the proof of the lemma.

Note that the privacy budget of TPSL only depends on the budget of \text{GradPerturb}, but not on the number of samples \( n \) or number of batches \( B \).

### 4.2 Improve privacy budget analysis

From the above proof, we see that if we can couple the generation of \( \tilde{g}_{b,i} \) and \( \tilde{u}_{b,i} \), then it is possible to further tighten the bound of privacy budget. In the setting of neural networks, we can achieve this by adding noise on the final hidden layer. Formally, let \( \theta^* \) be the parameters of the last hidden
layers of the sub-network $h_{\theta_i}(\cdot)$ held by $P_L$, then we can rewrite $h_{\theta_i}(\cdot)$ as $h_1(\theta^*, h_2(\tilde{\theta}_L, \cdot))$ where $\tilde{\theta}_L$ is the remaining parameters in $\theta_L$. From the chain rule of derivatives, we have

$$g_{b,i} = \frac{\partial \ell_{b,i}}{\partial e_{b,i}} = \frac{\partial (h_1(\theta^*, h_2(\tilde{\theta}_L, e_{b,i})), y)}{\partial e_{b,i}} = \frac{\partial h_1}{\partial h_1} \frac{\partial h_1}{\partial e_{b,i}},$$

$$u_{b,i} = \frac{\partial \ell_{b,i}}{\partial \theta_L} = \frac{\partial (h_1(\theta^*, h_2(\tilde{\theta}_L, e_{b,i})), y)}{\partial \theta_L} = \frac{\partial h_1}{\partial h_1} \frac{\partial h_1}{\partial \theta_L}.$$ 

Hence we can apply \texttt{GradPerturb} once on $\frac{\partial \ell_{b,i}}{\partial h_1}$, i.e., $P_L$ computes $v_{b,i,j} := \frac{\partial \ell_{b,i,j}}{\partial h_1}$ for $j \in \{0, 1\}$ and $\tilde{v}_{b,i} = \texttt{GradPerturb}(y_{b,i}, v_{b,i,0}, v_{b,i,1})$. Then $P_L$ computes $\tilde{g}_{b,i} = \tilde{v}_{b,i} \frac{\partial h_1}{\partial e_{b,i}}$ and $\tilde{u}_{b,i} = \tilde{v}_{b,i} \frac{\partial h_1}{\partial \theta_L}$. With similar proof, now we can show that when \texttt{GradPerturb} is $(\epsilon, 0)$-DP, we have

$$\Pr\left[\left|\left(S_t, S_m\right)(D)\right|_{S_* = [s]S_*} = \left|\left(S_t, S_m\right)(D')\right|_{S_* = [s]S_*}\right] \leq e^{\epsilon}.$$

To summarize, by slightly modifying Algorithm 3, we can improve the bound on privacy budget to $(\epsilon, 0)$ from $(2\epsilon, 0)$.

5 Experiments

In this section, we report the experiment results of split learning algorithms. We train a modified Wide&Deep model [Cheng et al., 2016] on two real-world large-scale datasets, Avazu [ava, 2015] and Criteo [cri, 2014]. The model $f$ held by the non-label party $P_N$ is the embedding layer for input features plus two 128-unit ReLU activated multi-layer perceptron (MLP) layers. The model $h$ held by the label party $P_L$ is two layers of 128-unit ReLU activated MLP. We use SGD with batch size $8, 192$ for training.

5.1 Label leakage in split learning

We first show that the naive implementation of split learning, for example Algorithm 1, does leak label information, which verify the findings of Li et al. [2022]. We consider three kinds of attack: norm attack (NA), spectral attack (SA) and shortest distance attack (SDA). Norm attack [Li et al., 2022] is a simple heuristic for black-box attack, and has been shown to be effective on imbalanced datasets like Avazu and Criteo. Spectral attack [Tran et al., 2018, Sun et al., 2021] is a 2-clustering algorithm that works as a black-box attack. More details about these attacks can be found in Appendix A. Shortest distance attack is the white-box attack introduced in Section 3.1.

We use attack AUC for the evaluation of these attacks. Detailed description of attack AUC is in Appendix A.3. Attack AUC close to 1 means the attack can almost perfectly reconstruct the labels, while close to 0.5 attack AUC means the attack behaves similarly to random guess.

We evaluate these attacks on Algorithm 1 with the Avazu dataset and report the results in Table 1. For reference we also report the test AUC. We can clearly see that all three attacks can recover the true label almost perfectly. Furthermore, white-box attack (i.e. shortest distance attack) achieves better performance compared to black-box attacks.

| test AUC | NA AUC | SA AUC | SDA AUC |
|----------|--------|--------|---------|
| 0.7523   | 0.9897 | 0.9890 | 1.0000  |

Table 1: Attacks on non-private split learning on Avazu.
5.2 Results of transcript DP split learning

We evaluate the performance of the two approaches we propose: Laplace perturbation (Laplace) and discrete perturbation (Discrete). For different choices of \( \epsilon \), we report test performance as the utility metric. We also report the attack AUC for the attacks introduced in Section 5.1. The results on Avazu are presented in Table 2, and the results on Criteo are presented in Table 3. For comparison, we also include a baseline of isotropic Gaussian perturbation (Gaussian). That is, we use \( \text{GradPerturb}(y, g_0, g_1) = g_y + r \) where \( r \) is an isotropic Gaussian vector. Notice that Gaussian mechanism cannot achieve pure DP (i.e. \( \delta = 0 \)), hence our framework of Lemma 4.1 does not apply and we do not report the value of \( \epsilon \) for Gaussian perturbation. We apply GradPerturb on the final hidden layer as described in Section 4.2.

We can see that shortest distance attack is more informative than the other two black-box attacks. For instance, when \( \epsilon = 1 \), the attack AUC of norm attack and spectral attack of Laplace perturbation is about 0.57, which seems to be low. However, this is a false sense of security: the shortest distance attack has attack AUC around 0.7.

To better illustrate the utility-privacy trade-off, we plot the curve of utility (measured by test AUC) vs. privacy (measured by privacy budget \( \epsilon \)) in Figure 1 (a)(c) for Laplace and Discrete. To compare with Gaussian, we use SDA as the privacy metric and draw Figure 1 (b)(d).

Table 2: Performance of different transcript DP split learning algorithms on Avazu.

| SDA AUC Level | Algorithms | SDA AUC | Test AUC | \( \epsilon \) | NA AUC | SA AUC |
|---------------|------------|---------|----------|-----------|--------|--------|
| Low           | Laplace    | 0.5237  | 0.6689   | 0.1       | 0.4932 | 0.4847 |
|               | Discrete   | 0.5274  | 0.6293   | 0.1       | 0.4997 | 0.4997 |
|               | Gaussian   | 0.5252  | 0.6657   | –         | 0.5117 | 0.4999 |
| Medium        | Laplace    | 0.6961  | 0.7416   | 1         | 0.5784 | 0.5743 |
|               | Discrete   | 0.7789  | 0.7342   | 1         | 0.4806 | 0.4796 |
|               | Gaussian   | 0.6455  | 0.7384   | –         | 0.5440 | 0.5363 |
| High          | Laplace    | 0.9622  | 0.7518   | 10        | 0.9640 | 0.9966 |
|               | Discrete   | 0.9999  | 0.7523   | 10        | 0.9898 | 0.9893 |
|               | Gaussian   | 1       | 0.7522   | –         | 0.9902 | 0.9899 |

Table 3: Performance of different transcript-DP split learning algorithms on Criteo.

| SDA AUC Level | Algorithms | SDA AUC | Test AUC | \( \epsilon \) | NA AUC | SA AUC |
|---------------|------------|---------|----------|-----------|--------|--------|
| Low           | Laplace    | 0.5236  | 0.7095   | 0.1       | 0.5011 | 0.5011 |
|               | Discrete   | 0.5258  | 0.6037   | 0.1       | 0.5138 | 0.5215 |
|               | Gaussian   | 0.5264  | 0.6614   | –         | 0.4985 | 0.5017 |
| Medium        | Laplace    | 0.6964  | 0.7639   | 1         | 0.5525 | 0.5525 |
|               | Discrete   | 0.7310  | 0.7497   | 1         | 0.7317 | 0.7284 |
|               | Gaussian   | 0.6409  | 0.7550   | –         | 0.5017 | 0.5031 |
| High          | Laplace    | 0.9965  | 0.7784   | 10        | 0.8761 | 0.8766 |
|               | Discrete   | 0.9999  | 0.7792   | 10        | 0.9202 | 0.9204 |
|               | Gaussian   | 1       | 0.7786   | –         | 0.9038 | 0.9159 |
We observe that Laplace and Gaussian have similar utility-privacy trade-off curves, which confirms the insight that to protect label privacy, it is sufficient to add noise in the direction of \(g_0 - g_1\). Furthermore, Laplace achieves good utility within reasonable range of \(\epsilon\). Compared to the non-private baseline, when \(\epsilon = 1\), the test AUC only drops by 1.4%.

We also observe that Discrete has worse utility-privacy trade-off than Laplace and Gaussian. One explanation is that Discrete generates biased estimators, namely \(E[\text{GradPerturb}(y, g_0, g_1)] \neq g_y\) for Discrete. However, the other two methods provide unbiased gradient estimator.

![Graphs showing trade-off of transcript DP algorithms on Avazu, trade-off measured by shortest attack AUC on Avazu, trade-off of transcript DP algorithms on Criteo, trade-off measured by shortest attack AUC on Criteo.](image)

Figure 1: Utility-privacy trade-off.

6 Extensions

6.1 Go beyond online learning

Our learning algorithm TPSL is an online training algorithm [Duchi et al., 2011] in the sense that data is accessed in a sequential order. Online learning is widely used in practice to train neural networks due to its scalability and efficiency. Our proof of differential privacy makes use of the property of online learning that each training sample is used only once in the entire procedure.
Algorithm 4: \texttt{GradPerturb} with multiple labels equipped with distribution $U$ over $\mathbb{R}^k$

| Line | Description |
|------|-------------|
| \textbf{Input:} | True label $y \in \{0, 1, \cdots, k-1\}$. Gradients $g_0, g_1, \cdots, g_{k-1}$ corresponds to label $0, 1, \cdots, k-1$ respectively. |
| 1 | $P_L$ samples $u \sim U$. |
| 2 | return $g_y + \sum_{i=0}^{k-1} u_i \cdot g_i$. |

However, our perturbation-based methods can also be used even if samples can be used multiple times, such as in multi-epoch training. The strategy of the label party $P_L$ needs to be modified slightly: when $P_L$ calls \texttt{GradPerturb} for one training sample for the first time, $P_L$ needs to record the random scalar generated by \texttt{GradPerturb}. Whenever the same sample is used again, instead of running a fresh \texttt{GradPerturb}, $P_L$ just uses the recorded random scalar to generate the perturbed gradient. It can be shown that this algorithm is still transcript differentially private.

6.2 Multi-label classification

Our learning scheme can be easily extended to multi-label classification. Notice that the proof of Lemma 4.1 does not depend on domain of labels $Y$ being $\{0, 1\}$. When $Y = \{0, 1, 2, \cdots, k-1\}$ with $k > 2$, as long as we can design differentially private perturbation scheme \texttt{GradPerturb}, then TPSL is still differentially private by Lemma 4.1.

To deal with multiple labels, we design Algorithm 4. It is a natural generalization of Algorithm 2. Instead of only adding noise in the direction of $g_{y_1} - y - g_y$ as in the binary case, we now add noise on each $g_{y'} - g_y$ for $y' = 0, 1, \cdots, k-1$. Notice that the output of Algorithm 2 can be interpreted as

$$g_y + \sum_{i=0}^{k-1} u_i \cdot g_i = (1 + u_y - \sum_{y' \neq y} u_{y'})g_y + \sum_{y' \neq y} u_{y'}(g_{y'} - g_y).$$

The first term $(1 + u_y - \sum_{y' \neq y} u_{y'})g_y$ is a scaled version of the correct gradient $g_y$. The second term $\sum_{y' \neq y} u_{y'}(g_{y'} - g_y)$ is just the noise added on the direction of each $g_{y'} - g_y$.

We give two instances of the distribution $U$ that makes Algorithm 4 $(\epsilon, 0)$-DP. As the consequence, TPSL is also $(2\epsilon, 0)$-DP when equipped with such $U$.

- (Multi-class Laplace perturbation) $U = (U_0, U_1, \cdots, U_{k-1})$ where $U_i$ are i.i.d sampled from $\text{Lap}(2/\epsilon)$.

- (Multi-class discrete perturbation) Let $e_i \in \mathbb{R}^d$ be the $i$-th (starting from 0) unit vector, that is, the $i$-th coordinate is 1 and all the rest coordinates are 0. Then $U$ is chosen from $\{e_i - e_y\}_{i=0}^{k-1}$ with probability

$$\Pr[U = e_i - e_y] = \begin{cases} e^i & \text{if } i = y, \\ e^{i+k-1} & \text{else.} \end{cases}$$

The DP guarantees are given in the following lemmas. The detailed proof is deferred to Appendix B.

**Lemma 6.1.** \texttt{GradPerturb} equipped with multi-class Laplace perturbation is $(\epsilon, 0)$-DP.

**Lemma 6.2.** \texttt{GradPerturb} equipped with multi-class discrete perturbation is $(\epsilon, 0)$-DP.
7 Related Works

**Differentially Private Machine Learning.** Differential privacy and machine learning are closely related [Dwork et al., 2015], and there has been a long line of research on designing differentially private machine learning models [Chaudhuri et al., 2011, Lei, 2011, Vaidya et al., 2013, Smith et al., 2018, Phan et al., 2020]. For the case of deep learning, since the breakthrough work of Abadi et al. [2016] of DP-SGD, there has been many research works on differentially private deep learning [Papernot et al., 2016, 2018, McMahan et al., 2018, Tramer and Boneh, 2020]. Recently, Ghazi et al. [2021] and Malek Esmaeili et al. [2021] studied the case where only labels are considered sensitive, and showed that with the same privacy budget, it can achieve better model performance compared to DP-SGD. In these works, differential privacy is usually enforced on the weights of machine learning models. This is different from our setting of Vertical Federated Learning, where the adversary can also access the shared messages. Hence we also need to make the communication differentially private.

**Differentially Private Vertical Federated Learning.** Prior to our work, researchers have explored using differential privacy in Vertical Federated Learning. For logistic regression, Wang et al. [2020] proposed using Gaussian mechanism on the intermediate results to improve running time while maintain privacy. For neural networks, Chen et al. [2020] added differentially private noise on each layer and showed that with sufficiently large noise, the privacy budget can be met. Compared with these works, our method can work with neural networks, and since we focus on protecting label privacy, no extra gradient clipping is needed, which is in contrast to Chen et al. [2020] or DP-SGD [Abadi et al., 2016].

8 Conclusion

In this work, we consider split learning in the multi-party computation setting and introduce the notion of transcript differential privacy to measure the label privacy. Inspired by the white-box attack, we propose a generic gradient-perturbation scheme GradPerturb that adds noise only on the optimal direction. Based on GradPerturb, we propose our split learning framework of TPSL. To the best of our knowledge, this is the first split learning algorithm that provides provable differential privacy guarantees. Experiments over large-scale datasets show that with Laplace perturbation, TPSL can achieve good utility-privacy trade-off.

It remains an open question that how we can build a split learning protocol so that the input features can also be protected. A straightforward implementation to enforce differential privacy on input features is to add noise not only on the gradients in the backward phase, but also on the embeddings in the forward phase. It is would be interesting if we can find smarter ways than adding isotropic noise. Another interesting question is that if we can build up connections between the privacy notions we introduce, including transcript DP and multiple attack AUCs.

References

Display advertising challenge. https://www.kaggle.com/c/criteo-display-ad-challenge/data, 2014.

Click-through rate prediction. https://www.kaggle.com/c/avazu-ctr-prediction/data, 2015.

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and
Li Zhang. Deep learning with differential privacy. In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pages 308–318, 2016.

John M Abowd. The us census bureau adopts differential privacy. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2867–2867, 2018.

Sharif Abuadbba, Kyuyeon Kim, Minki Kim, Chandra Thapa, Seyit A Camtepe, Yansong Gao, Hyoongshick Kim, and Surya Nepal. Can we use split learning on 1d cnn models for privacy preserving training? In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, pages 305–318, 2020.

Iker Ceballos, Vivek Sharma, Eduardo Mugica, Abhishek Singh, Alberto Roman, Praneeth Vepakomma, and Ramesh Raskar. Splitnn-driven vertical partitioning. arXiv preprint arXiv:2008.04137, 2020.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. Journal of Machine Learning Research, 12(3), 2011.

Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. Vafl: a method of vertical asynchronous federated learning. arXiv preprint arXiv:2007.06081, 2020.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In Proceedings of the 1st workshop on deep learning for recommender systems, pages 7–10, 2016.

Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. arXiv preprint arXiv:1712.01524, 2017.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research, 12(7), 2011.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci., 9(3-4):211–407, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Theory of cryptography conference, pages 265–284. Springer, 2006.

Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. In 2010 IEEE 51st Annual Symposium on Foundations of Computer Science, pages 51–60. IEEE, 2010.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In Proceedings of the forty-seventh annual ACM symposium on Theory of computing, pages 117–126, 2015.

Ege Erdogan, Alptekin Kupcu, and A Ercument Cicek. Unsplit: Data-oblivious model inversion, model stealing, and label inference attacks against split learning. arXiv preprint arXiv:2108.09033, 2021.

Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. Secure linear regression on vertically partitioned datasets. IACR Cryptol. ePrint Arch., 2016:892, 2016.
Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. On deep learning with label differential privacy. *arXiv preprint arXiv:2102.06062*, 2021.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19586–19597. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/e32cc80bf07915058ce90722ee17bb71-Paper.pdf.

Oktkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtarik. Lower bounds and optimal algorithms for personalized federated learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2304–2315. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/187acf7982f3c169b3075132380986e4-Paper.pdf.

Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*, 2017.

Miran Kim, Yongsoo Song, Shuang Wang, Yuhou Xia, Xiaojian Jiang, et al. Secure logistic regression based on homomorphic encryption: Design and evaluation. *JMIR medical informatics*, 6(2):e8805, 2018.

Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious prf with applications to private set intersection. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS ’16, page 818–829, 2016.

Jing Lei. Differentially private m-estimators. *Advances in Neural Information Processing Systems*, 24:361–369, 2011.

Oscar Li, Jiankai Sun, Xin Yang, Weihao Gao, Hongyi Zhang, Junyuan Xie, Virginia Smith, and Chong Wang. Label leakage and protection in two-party split learning. *International Conference on Learning Representations*, 2022.

Mani Malek Esmaeili, Ilya Mironov, Karthik Prasad, Igor Shilov, and Florian Tramer. Antipodes of label differential privacy: Pate and alibi. *Advances in Neural Information Processing Systems*, 34, 2021.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755*, 2016.
Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Ulfar Erlingsson. Scalable private learning with pate. In *International Conference on Learning Representations*, 2018.

Hai Phan, My T Thai, Han Hu, Ruoming Jin, Tong Sun, and Dejing Dou. Scalable differential privacy with certified robustness in adversarial learning. In *International Conference on Machine Learning*, pages 7683–7694. PMLR, 2020.

Benny Pinkas, Thomas Schneider, and Michael Zohner. Scalable private set intersection based on ot extension. *ACM Transactions on Privacy and Security (TOPS)*, 21(2):1–35, 2018.

Michael Smith, Mauricio Álvarez, Max Zwijssele, and Neil D Lawrence. Differentially private regression with gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 1195–1203. PMLR, 2018.

Jiankai Sun, Xin Yang, Yuanshun Yao, Aonan Zhang, Weihao Gao, Junyuan Xie, and Chong Wang. Vertical federated learning without revealing intersection membership. *arXiv preprint arXiv:2106.05508*, 2021.

Florian Tramer and Dan Boneh. Differentially private learning needs better features (or much more data). In *International Conference on Learning Representations*, 2020.

Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, 2018.

Jaideep Vaidya, Basit Shafiq, Anirban Basu, and Yuan Hong. Differentially private naive bayes classification. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 1, pages 571–576. IEEE, 2013.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

Chang Wang, Jian Liang, Mingkai Huang, Bing Bai, Kun Bai, and Hao Li. Hybrid differentially private federated learning on vertically partitioned data. *arXiv preprint arXiv:2009.02763*, 2020.

Kai Yang, Tao Fan, Tianjian Chen, Yuanming Shi, and Qiang Yang. A quasi-newton method based vertical federated learning framework for logistic regression. *arXiv preprint arXiv:1912.00513*, 2019a.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019b.

Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5332–5344. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/39d0a8908fbe6c18039ea8227f827023-Paper.pdf.
A  Label inference attacks from gradients

In this section, we introduce two black-box attacks, norm attack and spectral attack. We then introduce attack AUC as the evaluation metric for these attacks.

A.1 Norm attack

Norm attack [Li et al., 2022] is a simple heuristic attack that is found to be effective on imbalanced datasets. On gradient $g$, the output of norm attack is a scalar

$$NA(g) := \|g\|_2^2.$$  

It turns out that positive samples tend to have larger norm compared to negative samples. Hence norm attack can be used to recover the label information.

A.2 Spectral attack

Spectral attack is a singular value decomposition (SVD) based outlier detection method introduced by Tran, Li and Madry [Tran et al., 2018]. In particular, they show that

**Lemma A.1** (Lemma 3.1, Definition 3.1 in Tran et al. [2018]). Fix $0 < \epsilon < \frac{1}{2}$. Let $D, W$ be two distributions over $\mathbb{R}^d$ with mean $\mu_D, \mu_W$ and covariance matrices $\Sigma_D, \Sigma_W$. Let $F$ be a mixture distribution given by $F = (1 - \epsilon)D + \epsilon W$. If $\|\mu_D - \mu_W\|_2^2 \geq \frac{6\sigma^2}{\epsilon}$, then the following statement holds: let $\mu_F$ be the mean of $F$ and $v$ be the top singular vector of the covariance matrix of $F$, then there exists $t > 0$ so that

$$\Pr_{X \sim D}[\|X - \mu_F, v\| > t] < \epsilon,$$

$$\Pr_{X \sim W}[\|X - \mu_F, v\| < t] < \epsilon.$$  

Sun et al. [2021] shows that the above lemma can be used as a 2-clustering algorithm: let $D$ and $W$ be two distributions that we would like to distinguish, and we are given a collection of samples $F$ chosen from $D$ and $W$. Then we can use the value $|\langle X - \mu_F, v \rangle|$ to cluster the samples.

We use this idea to build a black-box attack, which we call spectral attack. In particular, we think of $F$ as the distribution of gradients of positive samples, and $W$ as the distribution of gradients of negative samples. In our experiment, we conduct spectral attack in every mini-batch. We estimate $\mu_F$ and $v$ by computing the empirical mean and covariance matrix with data in the mini-batch. Then the output of spectral attack is computed by

$$SA(g) := |\langle g - \mu_F, v \rangle|.$$  

A.3 Attack AUC

As the output of these attacks may not be Boolean, standard metrics like accuracy, precision and recall do not evaluate the effectiveness of the attacks well. In order to get rid of the problem of setting threshold for binary decision, we instead report the attack AUC (area under curve), which we introduce as follows. Fix a set of samples $(g_1, y_1), \cdots, (g_k, y_k)$. Let the output of some attack be $(s_1, \cdots, s_k)$. For different threshold $t$, we can turn the fractional output of the attack into Boolean decision, and obtain corresponding False Positive Rate (FPR) and True Positive Rate (TPR) by

$$FPR(t) := \frac{|i \in [k] : s_i \geq t, y_i = 0|}{|i \in [k] : y_i = 0|},$$

$$TPR(t) := \frac{|i \in [k] : s_i < t, y_i = 1|}{|i \in [k] : y_i = 1|}.$$  

17
Then the attack AUC is defined the area under the ROC (receiver operating characteristic) curve, which is

\[
\text{attack AUC} := \int_{-\infty}^{+\infty} TPR(t) dFPR(t).
\]

**B Proofs**

In this section we present proofs that are omitted in the main paper.

**B.1 Proof of Lemma 3.1**

*Proof of Lemma 3.1.* Fix \( g_0, g_1 \). We start with an auxiliary statement. Let the universe \( \mathcal{X} \subset \mathbb{R} \) be \{0, 1\}. Consider the identical mapping \( f(x) := x \) and the Laplace mechanism \( f^{DP} = f + r \) with \( r \sim \text{Lap}(b) \). Then the claim is that when \( b \geq \frac{1}{\epsilon} \), \( f^{DP} \) is \((\epsilon, 0)\)-DP. This is because the \( \ell_1 \) sensitive of \( f \) is 1, hence the DP guarantee follows from the property of Laplace mechanism [Dwork and Roth, 2014].

Now we consider the deterministic mapping \( h : \mathbb{R} \rightarrow \mathbb{R}^d \) defined as

\[
h(x) = x \cdot g_1 + (1 - x) \cdot g_0.
\]

Since \( f^{DP} \) is \((\epsilon, 0)\)-DP, and DP is immune to post processing, \( h(f^{DP}) \) is also \((\epsilon, 0)\)-DP. On the other hand, we have

\[
\begin{align*}
h(f^{DP}(0)) &= g_0 + r \cdot (g_1 - g_0), \\
h(f^{DP}(1)) &= g_1 - r \cdot (g_0 - g_1).
\end{align*}
\]

Since \( r \) is symmetric, \(-r\) is distributed identically to \( r \). Hence we conclude that \( h(f^{DP}) \) is the same as \( \text{GradPerturb} \), which completes the proof of the lemma.

**B.2 Proof of Lemma 3.2**

*Proof of Lemma 3.2.* When \( U = \text{Bern}(p) \), we have \( \Pr[\text{GradPerturb}(y, g_0, g_1) = g_y] = 1 - p \) and \( \Pr[\text{GradPerturb}(y, g_0, g_1) = g_{1-y}] = p \). Hence for all \( g \in \{g_0, g_1\} \) and \( y \in \{0, 1\} \), we have

\[
\frac{\Pr[\text{GradPerturb}(y, g_0, g_1) = g]}{\Pr[\text{GradPerturb}(1-y, g_0, g_1) = g]} \leq \frac{1 - p}{p},
\]

which follows from \( p \leq \frac{1}{2} \). This means \( \text{GradPerturb} \) is \((\ln \frac{1-p}{p}, 0)\)-DP.

**B.3 Proof of Lemma 6.1**

*Proof of Lemma 6.1.* We first prove an auxiliary statement: suppose that we have the universe \( \mathcal{X} := \{e_i\}_{i=0}^{k-1} \subset \mathbb{R}^k \) and the identity mapping \( f(x) = x \), then the randomized mechanism \( f^{DP} := f + (r_0, r_1, \ldots, r_{k-1}) \) is \((\epsilon, 0)\)-DP where \( r_i \) are i.i.d. sampled from \( \text{Lap}(2/\epsilon) \). This is because the \( \ell_1 \) sensitivity of \( f \) is 2.
We then consider the deterministic mapping \( h : \mathbb{R}^k \rightarrow \mathbb{R}^d \) given by

\[
h(x_0, x_1, \cdots, x_{k-1}) = \sum_{i=0}^{k-1} x_i g_i.
\]

It is easy to verify that \( h(f^{DP}) \) is just the output of GradPerturb. Hence by the post-processing property of differential privacy, we have that GradPerturb is \((\epsilon, 0)\)-DP.

**B.4 Proof of Lemma 6.2**

*Proof of Lemma 6.2.* With multi-class discrete perturbation, the output of GradPerturb is \( \tilde{g}_{\tilde{y}} \) for \( \tilde{y} \in \{0, 1, \cdots, k-1\} \). Furthermore, for any \( y, \tilde{y} \in \{0, 1, \cdots, k-1\} \) we have

\[
\frac{1}{e^\epsilon + k - 1} \leq \Pr[\text{GradPerturb}(y, g_0, g_1, \cdots, g_{k-1}) = \tilde{g}_{\tilde{y}}] \leq \frac{e^\epsilon}{e^\epsilon + k - 1}.
\]

Hence, for \( y \neq y' \),

\[
\frac{\Pr[\text{GradPerturb}(y, g_0, g_1, \cdots, g_{k-1}) = \tilde{g}_{\tilde{y}}]}{\Pr[\text{GradPerturb}(y', g_0, g_1, \cdots, g_{k-1}) = \tilde{g}_{\tilde{y}}]} \leq e^\epsilon,
\]

which completes the proof of the lemma. \(\square\)