How Much Privacy Does Federated Learning with Secure Aggregation Guarantee?

Ahmed Roushdy Elkordy*
aelkordy@usc.edu
University of Southern California
USA

Jiang Zhang*
jiangzha@usc.edu
University of Southern California
USA

Yahya H. Ezzeldin
yessa@usc.edu
University of Southern California
USA

Konstantinos Psounis
kpsounis@usc.edu
University of Southern California
USA

Salman Avestimehr
avestime@usc.edu
University of Southern California
USA

ABSTRACT

Federated learning (FL) has attracted growing interest for enabling privacy-preserving machine learning on data stored at multiple users while avoiding moving the data off-device. However, while data never leaves users' devices, privacy still cannot be guaranteed since significant computations on users' training data are shared in the form of trained local models. These local models have recently been shown to pose a substantial privacy threat through different privacy attacks such as model inversion attacks. As a remedy, Secure Aggregation (SA) has been developed as a framework to preserve privacy in FL, by guaranteeing the server can only learn the global aggregated model update but not the individual model updates.While SA ensures no additional information is leaked about the individual model update beyond the aggregated model update, there are no formal guarantees on how much privacy FL with SA can actually offer; as information about the individual dataset can still potentially leak through the aggregated model computed at the server. In this work, we perform a first analysis of the formal privacy guarantees for FL with SA. Specifically, we use Mutual Information (MI) as a quantification metric and derive upper bounds on how much information about each user's dataset can leak through the aggregated model update. When using the FedSGD aggregation algorithm, our theoretical bounds show that the amount of privacy leakage reduces linearly with the number of users participating in FL with SA. To validate our theoretical bounds, we use an MI Neural Estimator to empirically evaluate the privacy leakage under different FL setups on both the MNIST and CIFAR10 datasets. Our experiments verify our theoretical bounds for FedSGD, which show a reduction in privacy leakage as the number of users and local batch size grow, and an increase in privacy leakage as the number of training rounds increases. We also observe similar dependencies for the FedAvg and FedProx protocol.

KEYWORDS

Federated Learning, Secure Aggregation, Mutual Information, Formal Privacy Guarantee

1 INTRODUCTION

Federated learning (FL) has recently gained significant interest as it enables collaboratively training machine learning models over locally private data across multiple users without requiring the users to share their private local data with a central server [9, 24, 30]. The training procedure in FL is typically coordinated through a central server who maintains a global model that is frequently updated locally by the users over a number of iterations. In each training iteration, the server firstly sends the current global model to the users. Next, the users update the global model by training it on their private datasets and then push their local model updates back to the server. Finally, the server updates the global model by aggregating the received local model updates from the users.

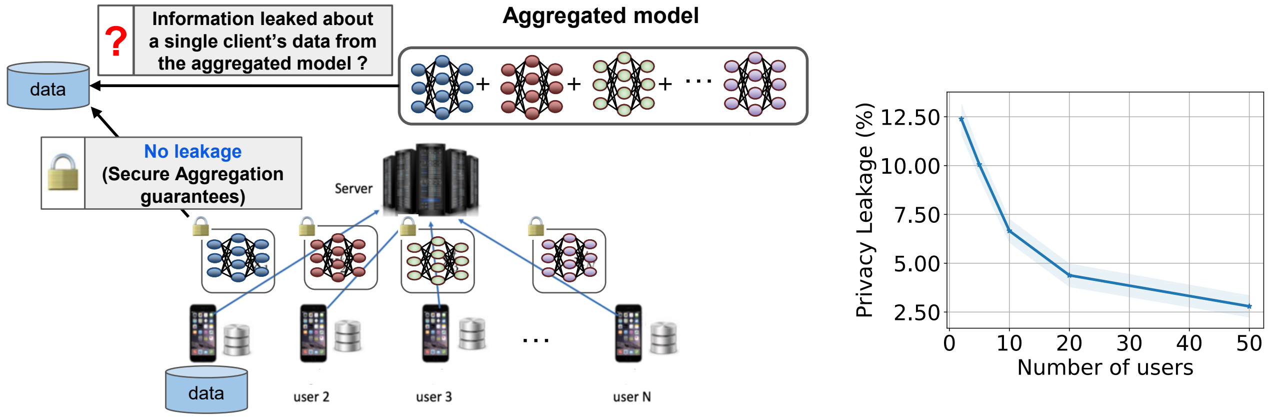
In the training process of FL, users can achieve the simplest notion of privacy in which users keep their data in-device and never share it with the server, but instead they only share their local model updates. However, it has been shown recently in different works (e.g., [18, 41, 44]) that this alone is not sufficient to ensure privacy, as the shared model updates can still reveal substantial information about the local datasets. Specifically, these works have empirically demonstrated that the private training data of the users can be reconstructed from the local model updates through what is known as the model inversion attack.

To prevent such information leakage from the individual models that are shared during the training process of FL, Secure Aggregation (SA) protocols have emerged as a remedy to address these privacy concerns by enabling the server to aggregate local model updates from a number of users, without observing any of their model updates in the clear. As shown in Fig. 1a, in each training round, users encrypt their local model updates before sending it to the server for aggregation. Thus, SA protocols formally guarantee that: 1) both the server and other users have no information about any user's clear model update from the encrypted update in the information theoretic sense; 2) the server only learns the aggregated model. In other words, secure aggregation ensures that only the aggregated model update is revealed to the server. Note that these SA guarantees allow for its use as a supporting protocol for other privacy-preserving approaches such as differential privacy [14]. In

[*] Both authors contributed equally to the paper.

Proceedings on Privacy Enhancing Technologies 2023(1), 510–526

https://doi.org/10.56553/popets-2023-0030



(a) Current and missing privacy guarantees for FL with secure aggregation

(b) Privacy leakage vs. number of users in FL with secure aggregation

Figure 1: Figure (a) illustrates the current formal privacy guarantee of FL with SA protocols and sheds light on the missing privacy guarantee on the aggregated model information leakage which is studied in this paper. Figure (b) gives a preview of the behavior of the privacy leakage through the global aggregated model for a CNN model as a function of the number of users in FL. The privacy leakage follows a $O(1/N)$ decay as proved in our theoretical bounds.

particular, these approaches can benefit from SA by reducing the amount of noise needed to achieve a target privacy level (hence improving the model accuracy) as demonstrated in different works (e.g., [23, 38]).

However, even with these SA guarantees on individual updates, it is not yet fully understood how much privacy is guaranteed in FL using SA, since the aggregated model update may still leak information about an individual user's local dataset. This observation leads us to the central question that this work addresses:

> *How much information does the aggregated model leak about the local dataset of an individual user?*

In this paper, we tackle this question by studying how much privacy can be guaranteed by using FL with SA protocols. We highlight that this work does not propose any new approaches to tackle privacy leakage but instead analyzes the privacy guarantees offered by state-of-the-art SA protocols, where updates from other users can be used to hide the contribution of any individual user. An understanding of this privacy guarantee may potentially assist other approaches such as differential privacy, such that instead of introducing novel noise to protect a user's model update, the randomized algorithm can add noise only to supplement the noise from other users' updates to the target privacy level. We can summarize the contributions of the work as follows.

Contributions. In this paper, we provide information-theoretic upper bounds on the amount of information that the aggregated model update (using FedSGD [9]) leaks about any single user's dataset under an honest-but-curious threat model, where the server and all users follow the protocol honestly, but can collude to learn information about a user outside their collusion set. Our derived upper bounds show that SA protocols exhibit a more favorable behavior as we increase the number of honest users participating in the protocol at each round. We also show that the information leakage from the aggregated model decreases by increasing the batch size, which has been empirically demonstrated in different recent works on model inversion attacks (e.g., [18, 41, 44]), where increasing the batch size limits the attack's success rate. Another interesting conclusion from our theoretical bounds is that increasing the model size does not have a linear impact on increasing the privacy leakage, but it depends linearly on the rank of the covariance matrix of the gradient vector at each user.

In our empirical evaluation, we conduct extensive experiments on the CIFAR10 [26] and MNIST [29] datasets in different FL settings. In these experiments, we estimate the privacy leakage using a mutual information neural estimator [6] and evaluate the dependency of the leakage on different FL system parameters: number of users, local batch size and model size. Our experiments show that the privacy leakage empirically follows similar dependencies to what is proven in our theoretical analysis. Notably, as the number of users in the FL system increase to 20, the privacy leakage (normalized by the entropy of a data batch) drops below 5% when training a CNN network on the CIFAR10 dataset (see Fig. 1b. We also show empirically that the dependencies, observed theoretically and empirically for FedSGD, also extend when using the FedAvg [9] FL protocol to perform multiple local training epochs at the users.

2 PRELIMINARIES

We start by discussing the basic federated learning model, before introducing the secure aggregation protocol and its state-of-the-art guarantees.

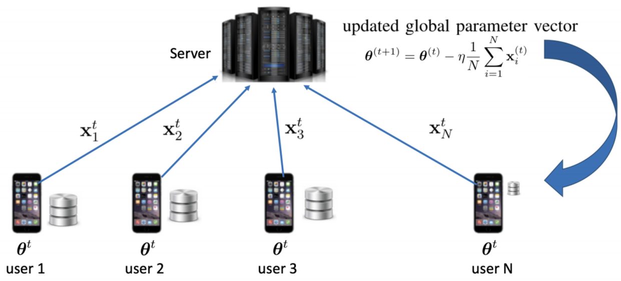


Figure 2: The training process in federated learning.

2.1 Basic Setting of Federated Learning

Federated learning is a distributed training framework [30] for machine learning, in which a set of users $\mathcal{N} = [N]$ ($|\mathcal{N}| = N$), each with its own local dataset $\mathcal{D}_i$ ($\forall i \in [N]$), collaboratively train a d-dimensional machine learning model parameterized by $\boldsymbol{\theta} \in \mathbb{R}^d$, based on all their training data samples. For simplicity, we assume that users have equal-sized datasets, i.e., $D_i = D$ for all $i \in [N]$. The typical training goal in FL can be formally represented by the following optimization problem:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \left[C(\boldsymbol{\theta}) := \frac{1}{N}\sum_{i=1}^{N} C_i(\boldsymbol{\theta}) \right], \tag{1}$$

where $\boldsymbol{\theta}$ is the optimization variable, $C(\boldsymbol{\theta})$ is the global objective function, $C_i(\boldsymbol{\theta})$ is the local loss function of user i. The local loss function of user i is given by

$$C_i(\boldsymbol{\theta}) = \frac{1}{D}\sum_{(x,y)\in\mathcal{D}_i} \ell_i(\boldsymbol{\theta}, (x, y)), \tag{2}$$

where $\ell_i(\boldsymbol{\theta}, (x, y)) \in \mathbb{R}$ denotes the loss function at a given data point $(x_i, y_i) \in \mathcal{D}_i$. The dataset $\mathcal{D}_i$ at user $i \in [N]$ is sampled from a distribution $\mathcal{P}_i$.

To solve the optimization problem in (1), an iterative training procedure is performed between the server and distributed users, as illustrated in Fig. 2. Specifically, at iteration t, the server firstly sends the current global model parameters, $\boldsymbol{\theta}^{(t)}$, to the users. User $i \in [N]$ then computes its model update $\mathbf{x}_i^{(t)}$ and sends it to the server. After that, the model updates of the N users are aggregated by the server to update the global model parameters into $\boldsymbol{\theta}^{(t+1)}$ for the next round according to

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta^{(t)}\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^{(t)}. \tag{3}$$

There are two common protocols for computing the model update $\mathbf{x}_i$: FedSGD and FedAvg [30]. Specifically, in FedSGD, each user uses a data batch $\mathcal{B}_i^{(t)}$ of size B sampled uniformly at random from it local dataset $\mathcal{D}_i$ to compute the model update as follows:

$$\mathbf{x}_i^{(t)} = \frac{1}{B}\sum_{b\in\mathcal{B}_i^{(t)}} g_i(\boldsymbol{\theta}^{(t)}, b), \tag{4}$$

where $g_i(\boldsymbol{\theta}^{(t)})$ is the stochastic estimate of the gradient $\nabla C_i(\boldsymbol{\theta}^{(t)})$ of the local loss function C_i of user i computed based on a random sample b (corresponding to (x_b, y_b)) drawn uniformly from $\mathcal{D}_i$ without replacement. In FedAvg, each user will run E complete local training rounds over its local dataset $\mathcal{D}_i$ to get its model update $\mathbf{x}_i^{(t)}$. Specifically, during each training round, each user will use all their mini-batches sampled from $\mathcal{D}_i$ to perform multiple stochastic gradient descent steps.

2.2 Secure Aggregation Protocols for Federated Learning

Recent works (e.g., [18, 41, 44]) have empirically shown that some of the local training data of user i can be reconstructed from the local model update $\mathbf{x}_i$, for $i \in [N]$. To prevent such data leakage, different SA protocols [3, 7, 13, 16, 22, 31, 35–38, 40, 43] have been proposed to provide a privacy-preserving FL setting without sacrificing the training performance. In the following, we discuss the threat model used in these SA protocols.

2.2.1 Threat Model in Secure Aggregation for Federated Learning. Most of SA protocols consider the honest-but-curious model [9] with the goal of uncovering users' data. In this threat model, the server and users honestly follow the SA protocol as specified. In particular, they will not modify their model architectures to better suit their attack, nor send malicious model update that do not represent the actually learned model. However, the server and the participating users are assumed to be curious and try to extract any useful information about the training data of any particular user. The extraction of the information is done by storing and analyzing the different data received during the execution of the protocol.

On the other hand, the threat model in theses SA protocols assumes that the server can collude with any subset of users $\mathcal{T} \subset [N]$ by jointly sharing any data that was used during the execution of the protocol (including their clear model updates $\mathbf{x}_i$, for all $i \in \mathcal{T}$) that could help in breaching the data privacy of any target user $i \in [N]/\mathcal{T}$. Similarly, this threat model also assumes that users can collude with each other to get information about the training data of other users.

2.2.2 Secure Aggregation Guarantees. In general, SA protocols that rely on different encryption techniques; such as homomorphic encryption [3, 13, 38, 40], and secure multi-party computing (MPC) [7, 16, 22, 31, 35–37, 43], are all similar in the encryption procedure in which each user encrypts its own model update $\mathbf{y}_i^{(t)} = \text{Enc}(\mathbf{x}_i^{(t)})$ before sending it to the server. This encryption is done such that these protocols achieve: 1) Correct decoding of the aggregated model under users' dropout; 2) Privacy for the local model update of the users from the encrypted model. In the following, we formally describe each of these guarantees.

Correct decoding. The encryption guarantees correct decoding for the aggregated model of the surviving users even if a subset $\mathcal{U} \subset [N]$ of the users dropped out during the protocol execution. In other words, the server should be able to decode

$$\text{Dec}\left(\sum_{i\in\mathcal{V}}\mathbf{y}_i^{(t)}\right) = \sum_{i\in\mathcal{V}}\mathbf{x}_i^{(t)}, \tag{5}$$

where $\mathcal{V}$ is the set of surviving users (e.g., $\mathcal{U} \cup \mathcal{V} = [N]$ and $\mathcal{U} \cap \mathcal{V} = \phi$).

Privacy guarantee. Under the collusion between the server and any strict subset of users $\mathcal{T} \subset [N]$, we have the following

$$I\left(\{\mathbf{y}_i^{(t)}\}_{i\in[N]}; \{\mathbf{x}_i^{(t)}\}_{i\in[N]} \middle| \sum_{i=1}^{N} \mathbf{x}_i^{(t)}, \mathbf{z}_{\mathcal{T}}\right) = 0, \tag{6}$$

where $\mathbf{z}_{\mathcal{T}}$ is the collection of information at the users in $\mathcal{T}$. In other words, (6) guarantees that under a given subset of colluding users $\mathcal{T}$ with the server, the encrypted model updates $\{\mathbf{y}_i^{(t)}\}_{i\in[N]}$ leak no information about the model updates $\{\mathbf{x}_i^{(t)}\}_{i\in[N]}$ beyond the aggregated model $\sum_{i=1}^{N} \mathbf{x}_i^{(t)}$. We note that the upper bound on the size of the colluding set $\mathcal{T}$ such that (6) is always guaranteed has been analyzed in the different SA protocols. Assuming that $|\mathcal{T}| \le \frac{N}{2}$ is widely used in most of the works (e.g., [36, 37]).

Remark 1. Recently, there have been also some works that enable doing secure model aggregation by using Trusted Execution Environments (TEE) such as Intel SGX (e.g., [28, 42]). SGX is a hardware-based security mechanism to protect applications running on a remote server. These TEE-based works are also designed to give the same guarantee in (6).

In the following, we formally highlight the weakness of the current privacy guarantee discussed in (6).

2.2.3 Our Contribution: Guarantees on Privacy Leakage from the Aggregated Model. Different SA protocols guarantee that the server doesn't learn any information about the local model update $\mathbf{x}_i^{(t)}$ of any user i from the received encrypted updates $\{\mathbf{y}_i^{(t)}\}_{i\in\mathcal{N}}$, beyond the aggregated model as formally shown in (6). However, it is not clear how much information the aggregated model update itself leaks about a single user's local dataset $\mathcal{D}_i$. In this work, we fill this gap by theoretically analyzing the following term.

$$I_{\text{priv/data}} = \max_{i\in[N]} I\left(\mathcal{D}_i; \left\{\frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i^{(t)}\right\}_{t\in[T]}\right). \tag{7}$$

The term in (7) represents how much information the aggregated model over T global training rounds could leak about the private data $\mathcal{D}_i$ of any user $i \in [N]$. In the following section, we theoretically study this term and discuss how it is impacted by the different FL system parameters such as model size, number of users , etc. In Section 5, we support our theoretical findings by empirically evaluating $I_{\text{priv/data}}$ in real-world datasets and different neural network architectures.

3 THEORETICAL PRIVACY GUARANTEES OF FL WITH SECURE AGGREGATION

In this section, we theoretically quantify the privacy leakage in FL when using secure aggregation with the `FedSGD` protocol.

3.1 Main Results

For clarity, we first state our main results under the honest-but-curious threat model discussed in Section 2.2.1 while assuming that there is no collusion between the server and users. We also assume that there is no user dropout. Later in Section 3.3, we discuss the general result with user dropout and the collusion with the server.

Our central result in this section characterizes the privacy leakage in terms of mutual information for a single round of `FedSGD`, which for round t is defined as

$$I_{\text{priv}}^{(t)} = \max_{i\in[N]} I\left(\mathbf{x}_i^{(t)}; \sum_{i=1}^{N} \mathbf{x}_i^{(t)} \middle| \left\{\sum_{i=1}^{N} \mathbf{x}_i^{(k)}\right\}_{k\in[t-1]}\right) \tag{8}$$

and then extends the privacy leakage bound to multiple rounds. Before stating our main result in Theorem 1 below, we first define two key properties of random vectors that will be used in stating our theorem and formally state our operational assumptions.

Definition 1 (Independent under whitening). *We say that a random vector* $\mathbf{v}$ *with mean* μ_v *and non-singular covariance matrix* $\mathbf{K}_v$ *is independent under whitening, if the whitened vector* $\widehat{\mathbf{v}}$ *is composed of independent random variables, where* $\widehat{\mathbf{v}} = \mathbf{K}_v^{-1/2}(\mathbf{v} - \mu_v)$.

Definition 2 (Uniformly σ-log concave). *A random vector* $\mathbf{v}$ *with covariance* $\mathbf{K}_v$ *is uniformly* σ*-log concave if it has a probability density function* $e^{-\phi(\mathbf{v})}$ *satisfying* $\nabla^2\phi(\mathbf{v}) \ge \mathbf{I}$ *and* $\exists\, \sigma > 0$, *such that* $\mathbf{K}_v \ge \sigma\mathbf{I}$.

Assumption 1 (IID data distribution). *Throughout this section, we consider the case where the local dataset* $\mathcal{Z}_i$ *are sampled IID from a common distribution, i.e., the local dataset of user i consists of IID data samples from a distribution* $\mathcal{P}_i$, *where* $\mathcal{P}_i = \mathcal{P}$ *for* $\forall i \in [N]$. *This implies that the distribution of the gradients* $g_i(\boldsymbol{\theta}^{(t)}, b)$, *for* $i \in [N]$, *conditioned on the last global model* $\boldsymbol{\theta}^{(t)}$ *is also IID. For this common conditional distribution, we will denote its mean with* $\mu_G^{(t)}$ *and the covariance matrix* $\mathbf{K}_G^{(t)}$ *in the t-th round.*

With the above definitions and using Assumption 1, we can now state our main result below, which is proved in Appendix A.

Theorem 1 (Single Round Leakage). *Let* $d^* \le d$ *be the rank of the gradient covariance matrix* $\mathbf{K}_G^{(t)}$, *and let* $\mathcal{S}_g$ *denote the set of subvectors of dimension* d^* *of* $g(\boldsymbol{\theta}^{(t-1)}, b)$ *that have a non-singular covariance matrices. Under Assumption 1, we can upper bound* $I_{\text{priv}}^{(t)}$ *for* `FedSGD` *in the following two cases:*

Case. 1 *If* $\exists \bar{g} \in \mathcal{S}_g$, *such that* $\bar{g}$ *is independent under whitening (see Def. 1), and* $\mathrm{E}|\bar{g}_i|^4 < \infty, \forall i \in [d^*]$, *then* $\exists\, C_{0,\bar{g}} > 0$, *such that*

$$I_{\text{priv}}^{(t)} \le \frac{C_{0,\bar{g}}\, d^*}{(N-1)B} + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right), \tag{9}$$

Case. 2 *If* $\exists \bar{g} \in \mathcal{S}_g$, *such that* $\bar{g}$ *is* σ*-log concave under whitening (see Def. 2) then we have that*

$$I_{\text{priv}}^{(t)} \le \frac{d^* C_{1,\bar{g}} - C_{2,\bar{g}}}{(N-1)B\sigma^4} + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right), \tag{10}$$

where: the constants $C_{1,\bar{g}} = 2\left(1 + \sigma + \log(2\pi) - \log(\sigma)\right)$ *and* $C_{2,\bar{g}} = 4\left(h(\bar{g}) - \frac{1}{2}\log(|\Sigma_{\bar{g}}|\right)$, *with* $\Sigma_{\bar{g}}$ *being the covariance matrix of the vector* $\bar{g}$.

Remark 2 (Simplified bound). Note that each $\bar{g} \in \mathcal{S}_g^{(t)}$ satisfying Case 1 or Case 2 gives an upper bound on $I_{\text{priv}}^{(t)}$. Let $\mathcal{S}_{g,c}^{(t)}$ be the set of $\bar{g} \in \mathcal{S}_g^{(t)}$ satisfying either Case 1 or Case 2. Then, we can combine these different bounds in Theorem 1 as follows

$$I_{\text{priv}}^{(t)} \leq \frac{d^*}{2}\log\left(\frac{N}{N-1}\right) + \frac{\min\limits_{\bar{g}\in\mathcal{S}_{g,c}^{(t)}}\left\{d^*\widehat{C}_{1,\bar{g}} - \widehat{C}_{2,\bar{g}}\right\}}{(N-1)B}, \tag{11}$$

where

$$(\widehat{C}_{1,\bar{g}}, \widehat{C}_{2,\bar{g}}) = \begin{cases} \left(C_{0,\bar{g}}, 0\right), & \text{if } \bar{g} \text{ satisfies Case 1,} \\ \left(\frac{C_{1,\bar{g}}}{\sigma^4}, \frac{C_{2,\bar{g}}}{\sigma^4}\right), & \text{if } \bar{g} \text{ satisfies Case 2,} \end{cases}$$

where $C_{0,\bar{g}}, C_{1,\bar{g}}$ and $C_{2,\bar{g}}$ are defined as in Theorem 1.

Remark 3. (Why the IID assumption?) Our main result in Theorem 1 relies on recent results on the entropic central [8, 15] for the sum of independent and identically random variables/vectors. Note that the IID assumption in the entropic central limit theorem can be relaxed to independent (but not necessarily identical) distributions, however, in this case, the upper bound will have a complex dependency on the moments of the N distributions in the system. In order to high-light how the privacy guarantee depends on the different system parameters (discussed in the next subsection), we opted to consider the IID setting in our theoretical analysis.

Remark 4. (Independence under whitening) One of our key assumptions in Theorem 1 is the independence under whitening assumption for stochastic gradient descent (SGD). This assumption is satisfied if the SGD vector can be approximated by a distribution with independent components or by a multivariate Gaussian vector. Our adoption of this assumption is motivated by recent theoretical results for analyzing the behaviour of SGD. These results have demonstrated great success in approximating the practical behaviour of SGD, in the context of image classification problems, by modeling the SGD with (i) a non-isotropic Gaussian vector [45], or, (ii) α-stable random vectors with independent components [34]. For both these noise models, the independence under whitening assumption in Theorem 1 is valid. However, a key practical limitation for the aforementioned SGD models (and thus of the independence under whitening assumption) is assuming a smooth loss function for learning. This excludes deep neural networks that make use of non-smooth activation and pooling functions (e.g., ReLU and max-pooling).

Now using the bounds in Theorem 1, in the following corollary, we characterize the privacy leakage of the local training data $\mathcal{D}_i$ of user i after T global training rounds of FedSGD, which is defined as

$$I_{\text{priv/data}} = \max_{i\in[N]} I\left(\mathcal{D}_i; \left\{\frac{1}{N}\sum_{i\in[N]} \mathbf{x}_i^{(t)}\right\}_{t\in[T]}\right), \tag{12}$$

Corollary 1. *Assuming that users follow the* FedSGD *training protocol and the same assumptions in Theorem 1, we can derive the upper bound of the privacy leakage $I_{\text{priv/data}}$ after T global training rounds of* FedSGD *in the following two cases:*
Case. 1: *Following the assumptions used in Case 1 in Theorem 1, we get*

$$I_{\text{priv/data}} \leq T\left[\frac{C_{0,\bar{g}}d^*}{(N-1)B} + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right)\right], \tag{13}$$

Case. 2: *Following the assumptions used in Case 2 in Theorem 1, we get*

$$I_{\text{priv/data}} \leq T\left[\frac{d^*C_{1,\bar{g}} - C_{2,\bar{g}}}{(N-1)B\sigma^4} + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right)\right]. \tag{14}$$

We prove Corollary 1 in Appendix B. Note that, we can combine the bounds in Corollary 1 similar to the simplification in (11) from Theorem 1.

3.2 Impact of System Parameters

3.2.1 Impact of Number of Users (N). As shown in Theorem 1 and Corollary 1, the upper bounds on information leakage from the aggregated model update decrease in the number of users N. Specifically, the leakage dependency on N is at a rate of $O(1/N)$.

3.2.2 Impact of Batch Size (B). Theorem 1 and Corollary 1 show that the information leakage from the aggregated model update could decrease when increasing the batch size that is used in updating the local model of each user.

3.2.3 Impact of Model Size (d). Given our definition of d^* in Theorem 1, where d^* represents the rank of the covariance matrix $K_{G^{(t)}}$ and $d^* \leq d$ (d is the model size), the leakage given in Theorem 1 and Corollary 1 only increases with increasing the rank of the covariance matrix of the gradient. This increase happens at a rate of $O(d^*)$. In other words, increasing the model size d (especially when the model is overparameterized) does not have a linear impact on the leakage. The experimental observation in Section 4 supports these theoretical findings.

3.2.4 Impact of Global Training Rounds (T). Corollary 1 demonstrates that the information leakage from the aggregated model update about the private training data of the users increases with increasing the number of global training rounds. This result reflects the fact as the training proceed, the model at the server start to memorize the training data of the users, and the data of the users is being exposed multiple times by the server as T increases, hence the leakage increases. The increase of the leakage happens at a rate of $O(T)$.

3.3 Impact of User Dropout, Collusion, and User Sampling

In this section, we extend the results given in Theorem 1 and Corollary 1 to cover the more practical FL scenario that consider, user dropout, the collusion between the server and the users and user sampling. We start by discussing the impact of user dropout and collusion.

3.3.1 Impact of User Dropout and Collusion with the Server. Note that, in the case of user dropouts, this is equivalent to a situation where the non-surviving users send a deterministic update of zero. As a result, their contribution can be removed from the aggregated model, and we can, without loss of generality, consider an FL system where only the surviving subset $\mathcal{N}_s \subset [N]$ users participate in the system.

Similarly, when a subset of users colludes with the server, then the server can subtract away their contribution to the aggregated model in order to unmask information about his target user i. As a result, we can again study this by considering only the subset of non-colluding (and surviving, if we also consider dropout) users in our analysis. This observation gives us the following derivative of the result in Theorem 1 which can summarized by the following corollary.

Corollary 2. *In* FedSGD*, under the assumptions used in Theorem 1, if there is only a subset $\mathcal{N}_s^{(t)} \subset [N]$ of non-colluding and surviving users in the global training round t, then, we have the following bound on $I_{\text{priv}}^{(t)}$*

$$I_{\text{priv}}^{(t)} \le \frac{d^*}{2} \log\left(\frac{|\mathcal{N}_s|}{|\mathcal{N}_s|-1}\right) + \frac{\min\limits_{\bar{g} \in \mathcal{S}_{g,c}^{(t)}} \left\{ d^* \widehat{C}_{1,\bar{g}} - \widehat{C}_{2,\bar{g}} \right\}}{(|\mathcal{N}_s|-1)B}, \tag{15}$$

where the maximization in $I_{\text{priv}}^{(t)}$ (given in (8)*) is only over the set of non-colluding surviving and non-colluding users; and the constants $\widehat{C}_{1,\bar{g}}$ and $\widehat{C}_{2,\bar{g}}$ are given in Remark 2.*

This implies that the per round leakage increases when we have a smaller number of surviving and non-colluding users. Similarly, we can modify the bound in Corollary 1 to take into account user dropout and user collusion by replacing N with $|\mathcal{N}_s|$.

3.3.2 Impact of User Sampling. In Theorem 1 and Corollary 1, we assume that all N users in the FL system participate in each training round. If instead K users are chosen each round, then all leakage upper bound will be in terms of K, the number of users in each round, instead of N. Furthermore, through Corollary 1, we can develop upper bounds for each user i, depending on the number of rounds T_i that the user participated in. For example, taking into account selecting K users in each round denoted by $\mathcal{K}^{(t)}$, then the upper bound in (13) is modified to give the following information leakage for user i

$$\begin{aligned} I_{\text{priv/data}}(i) &= I\left(\mathcal{D}_i; \left\{ \frac{1}{K} \sum_{i \in \mathcal{K}^{(t)}} \mathbf{x}_i^{(t)} \right\}_{t \in [T]} \right) \\ &\le T_i \left[\frac{C_{0,\bar{g}} d^*}{(K-1)B} + \frac{d^*}{2} \log\left(\frac{K}{K-1}\right) \right], \end{aligned} \tag{16}$$

where $T_i = K/N$ if the set of K users are chosen independently and uniformly at random in each round.

Thus user sampling would improve the linear dependence of the leakage on T (Section 3.2.4), but increase the per round leakage due to a smaller number of users in each round (Section 3.2.1).

4 EXPERIMENTAL SETUP

4.1 MI Estimation

In order to estimate the mutual information in our experiments, we use Mutual Information Neural Estimator (MINE) which is the state-of-the-art method [6] to estimate the mutual information between two random vectors (see Appendix D for more details). In our experiments, at the t-th global training round, we use MINE to estimate $I(\mathbf{x}_i^{(t)}; \sum_{i=1}^N \mathbf{x}_i^{(t)} | \boldsymbol{\theta}^{(t-1)})$, i.e., the mutual information between model update of the i-th user $\mathbf{x}_i^{(t)}$ and the aggregated model update from all users $\sum_{i=1}^N \mathbf{x}_i^{(t)}$. Our sampling procedure is described as follows: 1) at the beginning of the global training round t, each user will first update its local model parameters as the global model parameters $\boldsymbol{\theta}^{(t-1)}$. 2) Next, each user shuffles its local dataset. 3) Then, each user will pick a single data batch from its local dataset (if using FedSGD) or use all local data batches (if using FedAvg) to update its local model. 4) Lastly, secure aggregation is used to calculate the aggregated model update. We repeat the above process for K times to get K samples $\{(\mathbf{x}_{i,k}^{(t)}; \sum_{i=1}^N \mathbf{x}_{i,k}^{(t)})\}_{k=1}^{k=K}$, where $\mathbf{x}_{i,k}^{(t)}$ represents the model update from the i-th user in the k-th sampling and $\sum_{i=1}^N \mathbf{x}_{i,k}^{(t)}$ represents the aggregated model update from the i-th user in the k-th sampling. Note that we use the $K-th$ (last) sample $\sum_{i=1}^N \mathbf{x}_{i,K}^{(t)}$ to update the global model.

We repeat the end-to-end training and MI estimation multiple times in order to get multiple MI estimates for each training round t. We use the estimates for each round to report the average MI estimate and derive the confidence interval (95%) for the MI estimation[1].

Lastly, when using MINE to estimate MI, we use a fully-connected neural network with two hidden layers each having 100 neurons each as T_θ (see Appendix D for more details) and we perform gradient ascent for 1000 iterations to train the MINE network.

4.2 Datasets and Models

Datasets. We use MNIST and CIFAR10 datasets in our experiments. Specifically, the MNIST dataset contains 60,000 training images and 10,000 testing images, with 10 classes of labels. The CIFAR10 dataset contains 50,000 training images and 10,000 testing images, with 10 classes of labels. For each of the dataset, we randomly split the training data into 50 local datasets with equal size to simulate a total number of 50 users with identical data distribution. Note that we describe how to generate users with non-identical data distribution when we evaluate the impact of user heterogeneity in Section 5.6.

Moreover, we use MINE to measure the entropy of an individual image in each of these datasets, as an estimate of the maximal potential MI privacy leakage per image. We report that the entropy of an MNIST image is 567 (bits) and the entropy of a CIFAR10 image is 1403 (bits). Note that we will use the entropy of training data to normalize the measured MI privacy leakage in Section 5.

Models. Table 1 reports the models and their number of parameters used in our evaluation. For MNIST dataset, we consider three different models for federated learning. For each of these models, it takes as input a 28×28 image and outputs the probability of 10 image classes. We start by using a simple linear model, with a dimension of 7850. Next, we consider a non-linear model with the same amounts of parameters as the linear model. Specifically, we use a single layer perceptron (SLP), which consists of a linear layer and a ReLU activation function (which is non-linear). Finally, we choose a multiple layer perceptron (MLP) with two hidden layers, each of which contains 100 neurons. In total, it has 89610 parameters. Since the MLP model we use can already achieve more than

[1] During our experiments, we observe that the estimated MI does not change significantly across training rounds. Hence, we average the estimated MI across training rounds when reporting our results.

Models for MNIST			
Name	Linear	SLP	MLP
Size (d)	7850	7850	89610
Models for CIFAR10			
Name	Linear	SLP	CNN
Size (d)	30730	30730	82554

Table 1: Models used for MNIST and CIFAR10 datasets. Note that SLP, MLP, and CNN represent Single Layer Perceptron, Multiple Layer Perceptron, and Convolutional Neural Network, respectively.

95% testing accuracy on MNIST dataset, we do not consider more complicated model for MNIST.

For the CIFAR10 dataset, we also evaluate three different models for FL. For each of these models, it will take as input an 32×32×3 image and outputs the probability of 10 image classes. Similar to MNIST, the first two models we consider are a linear model and a single layer perceptron (SLP), both of which contains 30720 parameters. The third model we consider is a Convolutional Neural Network (CNN) modified from AlexNet [27], which contains a total of 82554 parameters and is able to achieve a testing accuracy larger than 60% on CIFAR. We do not consider larger CNN models due to the limited computation resources.

5 EMPIRICAL EVALUATION

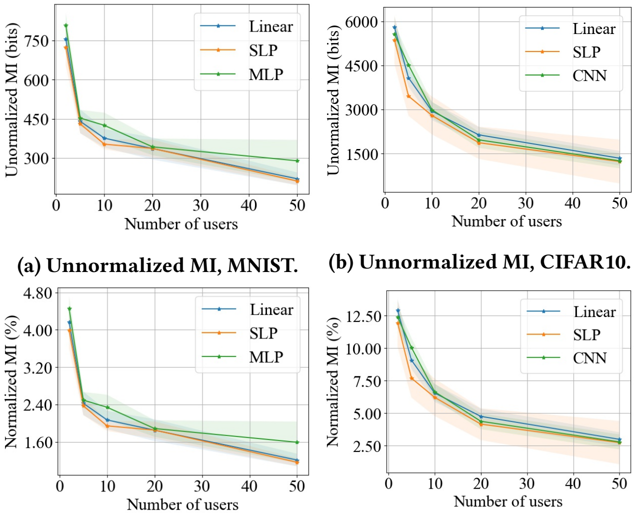
In this section, we empirically evaluate how different FL system parameters affect the MI privacy leakage in SA. Our experiments explore the effect of the system parameters on FedSGD, FedAvg and FedProx [33]. Note that our evaluation results on FedSGD are backed by our theoretical results in Section 3, while our evaluation results on FedAvg and FedProx are purely empirical.

We start by evaluating the impact of the number of users N on the MI privacy leakage for FedSGD, FedAvg and FedProx (see in Section 5.1). Then, we evaluate the impact of batch size B on the MI privacy leakage for both FedSGD, FedAvg and FedProx (see in Section 5.3). Next, in Section 5.4, we measure the accumulative MI privacy leakage across all global training rounds. We evaluate how the local training rounds E for each user will affect the MI privacy leakage for FedAvg and FedProx in Section 5.5. Finally, the impact of user heterogeneity on the MI privacy leakage for FedAvg is evaluated in Section 5.6.

We would like to preface by noting that FedProx differs from FedAvg by adding a strongly-convex proximal term to the loss used in FedAvg. Thus, we expect similar dependencies on the number of users N, batch-size B and local epochs E, when using FedAvg and FedProx.

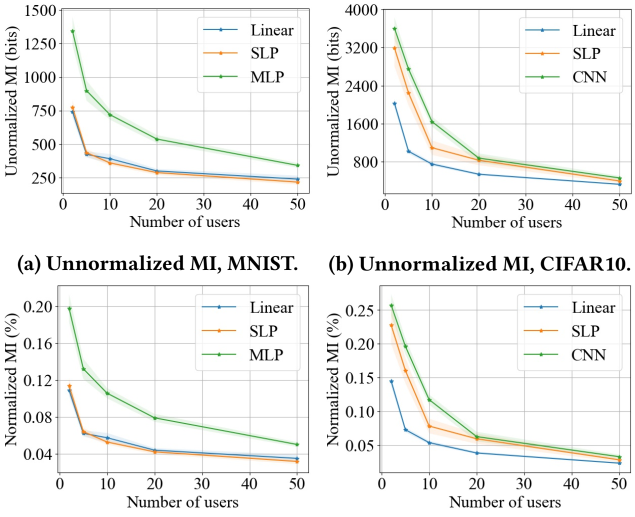
5.1 Impact of Number of Users (N)

FedSGD. Fig. 3 shows the impact of varying N on MI privacy leakage in FedSGD, where the number of users is chosen from $\{2, 5, 10, 20, 50\}$, and we measure the MI privacy leakage of different models on both MNIST and CIFAR10 datasets. We observe that increasing the number of users participating in FL using FedSGD will decrease the MI privacy leakage in each global training round (see Fig. 3a and 3b), which is consistent with our theoretical analysis in Section 3.2.1. Notably, as demonstrated in Fig. 3c and 3d, the percentile of MI privacy leakage (i.e. normalized by the entropy of a data batch) can drop below 2% for MNIST and 5% for CIFAR10 when there are more than 20 users.



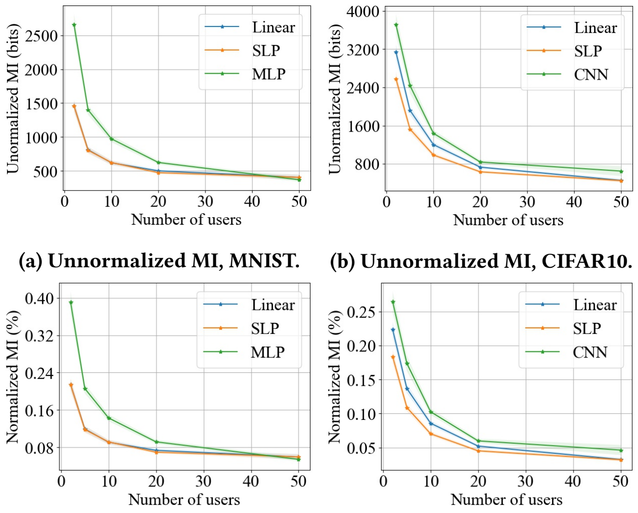
(a) Unnormalized MI, MNIST.
(b) Unnormalized MI, CIFAR10.
(c) Normalized MI, MNIST.
(d) Normalized MI, CIFAR10.

Figure 3: Impact of the number of users (N) when using FedSGD. Note that we set and $B = 32$ for all users on both MNIST and CIFAR10 datasets. We normalize the MI by entropy of a single data batch (i.e. $32*567$ for MNIST and $32*1403$ for CIFAR10).



(a) Unnormalized MI, MNIST.
(b) Unnormalized MI, CIFAR10.
(c) Normalized MI, MNIST.
(d) Normalized MI, CIFAR10.

Figure 4: Impact of the number of users (N) when using FedAvg. Note that we set E=1 and $B = 32$ for all users on both MNIST and CIFAR10 datasets. We normalize the MI by entropy of the whole local training dataset (i.e. $1200 * 567$ for MNIST and $1000 * 1403$ for CIFAR10).



(a) Unnormalized MI, MNIST. **(b) Unnormalized MI, CIFAR10.**

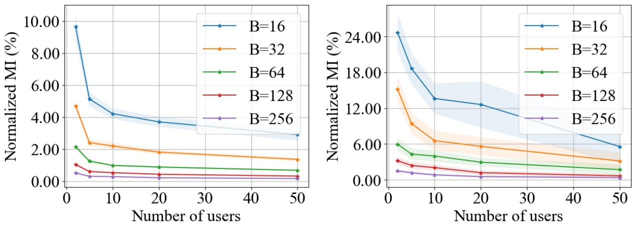
(c) Normalized MI, MNIST. **(d) Normalized MI, CIFAR10.**

Figure 5: Impact of the number of users (N) when using FedProx. Note that we set E=1 and $B = 32$ for all users on both MNIST and CIFAR10 datasets. We normalize the MI by entropy of a single data batch (i.e. $1200 * 567$ for MNIST and $1000 * 1403$ for CIFAR10).

FedAvg. Fig. 4 shows the impact of varying N on MI privacy leakage in FedAvg. Similar to the results in FedSGD, as the number of users participating in FedAvg increases, the MI privacy leakage in each global training round will decrease (see Fig. 4a and 4b), and the decreasing rate is approximately $O(N)$. Moreover, as shown in Fig. 4c and 4d, the percentile of MI privacy leakage drops below 0.1% on both MNIST and CIFAR10 when there are more than 20 users participating in FL. It is worth noting that we normalize the MI by the entropy of the whole training dataset in FedAvg instead of the entropy of a single batch, since users will iterate over all their data batches to calculate their local model updates in FedAvg. Therefore, although we observe that the unnormalized MI is comparable for FedSGD and FedAvg, the percentile of MI privacy leakage in FedAvg is significantly smaller than that in FedSGD.

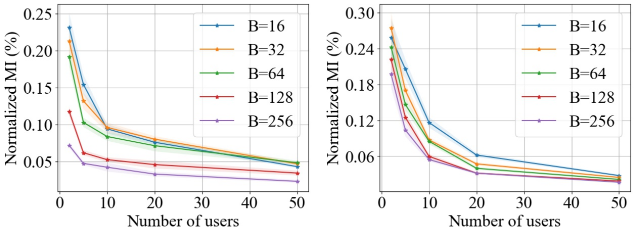
FedProx. Similar to FedAvg, Fig. 5 shows how the MI privacy leakage with FedProx varies with the number of users N. As the number of users increase, the MI privacy leakage decreases in each training round at an approximate rate of $O(N)$. With more than 20 participating users, the percentile of MI leakage drops below 0.12% under both MNIST and CIFAR10. Same as FedAvg, we normalize the MI privacy leakage by the entropy of the whole training dataset of a single user.

In conclusion, while our theoretical analysis on the impact of N in Section 3.2.1 is based on the assumption that the FedSGD protocol is used, our empirical study shows that it holds not only in FedSGD but also in FedAvg and FedProx.



(a) Normalized MI, MNIST. **(b) Normalized MI, CIFAR10.**

Figure 6: Impact of batch size (B) when using FedSGD. The MI is normalized by the entropy of a data batch, which is proportional to the batch size B (i.e. $B * 567$ for MNIST and $B * 1403$ for CIFAR10).



(a) Normalized MI, MNIST. **(b) Normalized MI, CIFAR10.**

Figure 7: Impact of batch size (B) when using FedAvg. The MI is normalized by the entropy of a user's local dataset, which is a constant (i.e. $1200 * 567$ for MNIST and $1000 * 1403$ for CIFAR10).

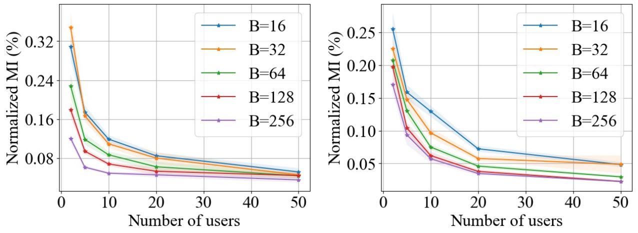
5.2 Impact of Model Size (d)

FedSGD. From Fig. 3, we observe that increasing model size d will increase the MI leakage during each global training round. However, the increase rate of MI leakage is smaller than the increase rate of d. This is expected since the upper bound of MI privacy leakage is proportional to d^* (i.e. the rank of the covariance of matrix as proved in Theorem 1), which will not increase linearly with d especially for overparameterized neural networks (see Section 3.2.3). Finally, we observe that the MI privacy leakage on CIFAR10 is generally higher than that on MNIST. Since the input images on CIFAR10 have higher dimension than the images on MNIST, larger model size are required during training. Therefore, we expect that the MI privacy leakage on CIFAR10 is higher than that on MNIST.

FedAvg and FedProx. As shown in Fig. 4 and Fig. 5, increasing the model size will also have a sub-linear impact on the increase of the MI privacy leakage in FedAvg and FedProx, which is consistent with our results in FedSGD.

5.3 Impact of Batch Size (B)

FedSGD. Fig. 6 shows the impact of varying B on the normalized MI privacy leakage in FedSGD, where the batch size is chosen from $\{16, 32, 64, 128, 256\}$ and we use MLP model on MNIST and CNN model on CIFAR10 during experiments. Note that we normalize the MI by the entropy of a single data batch used in each training round, which is proportional to the batch size B. On both MNIST and CIFAR10 datasets, we consistently observe that increasing B will decrease the MI privacy leakage in FedSGD, and the decay rate of MI is inversely proportional to batch size B. As demonstrated in Fig. 6, when there are more than 20 users, the percentile of MI privacy leakage for a single training round can be around 4% on



(a) Normalized MI, MNIST. **(b) Normalized MI, CIFAR10.**

Figure 8: Impact of batch size (B) when using FedProx. The MI is normalized by the entropy of a user's local dataset, which is a constant (i.e. $1200 * 567$ for MNIST and $1000 * 1403$ for CIFAR10).

MNIST and 12% on CIFAR10 with batch size 16. However, such leakage can drop to less 1% on both MNIST and CIFAR10 with batch size 256, which is significantly reduced.

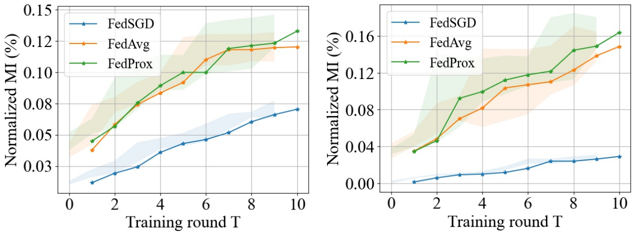
FedAvg and FedProx. Fig. 7 and Fig. 8 show the impact of varying the batch size B on MI privacy leakage in FedAvg and FedProx, respectively, following the same experimental setup as in Fig. 6. Since in both FedAvg and FedProx, each user will transverse their whole local dataset in each local training round, we normalize the MI by the entropy of the target user's local training dataset. As shown in Fig. 7 and Fig. 8, the impact of B in FedAvg and FedProx is relatively smaller than that in FedSGD. However, we can still observe that increasing B can decrease the MI privacy leakage in both FedAvg and FedProx. For example, with 20 users participating in FedAvg, the percentile of MI privacy leakage at each training round can drop from 0.8% to 0.3% when the batch size increases from 16 to 256, achieving a reduction in privacy leakage by a factor of more than 2×. Similarly, in FedProx, this causes a decrease in the MI privacy leakage from 0.09% to 0.04% when the batch size increases from 16 to 256.

In conclusion, we observe that increasing the batch size B can decrease the MI privacy leakage from the aggregated model update in FedSGD, FedAvg and FedProx which verifies our theoretical analysis in Section 3.2.3.

5.4 Accumulative MI leakage

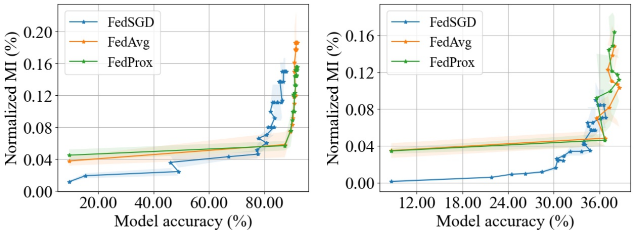
To evaluate how the accumulative MI privacy leakage will accumulate with the number of training round T, we measure the MI between training data and the aggregated model updates across training round. Specifically, given a local training dataset sample $\mathcal{D}_i$, we will concatenate the aggregated model updates $\{\frac{1}{N}\sum_{i\in\mathcal{N}}\mathbf{x}_i^{(t)}\}_{t\in[T]}$ across T training rounds in a single vector with dimension $d * T$. By randomly generating $\mathcal{D}_i$ for the target user for K times, we can get K concatenated aggregated model update vectors. Then, we use MINE to estimate $I(\mathcal{D}_i; \{\frac{1}{N}\sum_{i\in\mathcal{N}}\mathbf{x}_i^{(t)}\}_{t\in[T]})$ with these K dataset and concatenated model update samples.

As illustrated in Fig. 9, the MI privacy leakage will accumulate linearly as we increase the global training round T on both MNIST and CIFAR dataset, which is consistent with our theoretical results in Section 3.2.4. That also says, by reducing the times of local model aggregation, the MI privacy leakage of secure aggregation will be reduced. In practice, we can consider using client sampling to reduce the participation times of each client in FL, such that the accumulative MI leakage of individual users can be reduced. Moreover, we can also consider increasing the number of local averaging as much as possible to reduce the aggregation times for local model updates.



(a) Normalized accumulative MI, MNIST. **(b) Normalized accumulative MI, CIFAR10.**

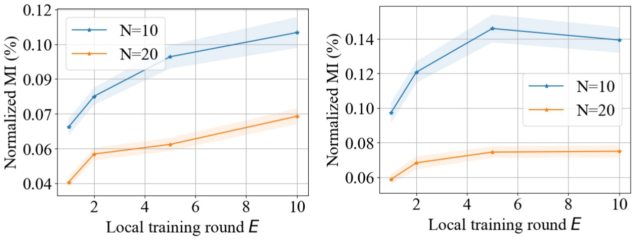
Figure 9: Accumulative MI privacy leakage on MNIST and CIFAR10 datasets. Note that we normalize the MI by the entropy of each user's local dataset, which will not change with T. We use the linear model for both MNIST and CIFAR10 datasets.



(a) MNIST **(b) CIFAR**

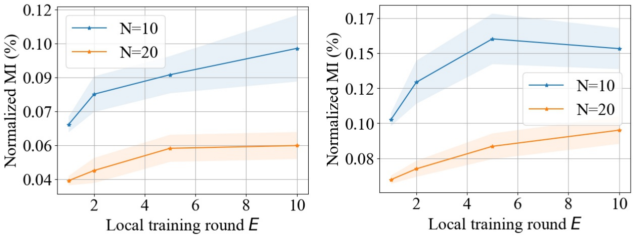
Figure 10: Accumulative MI privacy leakage vs model accuracy of different FL algorithms. Note that we use a linear model for case study and normalize the MI by the entropy of each user's local dataset.

Although, the three aggregation algorithms exhibit a similar trend with T, these algorithms can result in different convergence speeds to a target accuracy. To highlight the effect of convergence rate on the accumulative MI privacy leakage, we show, in Fig. 10, how the accuracy changes with the amount of MI leakage incurred for the three algorithms during the training process up to a maximum of 30 training rounds for FedSGD. We observe that although FedSGD achieves lower MI leakage for a fixed number of rounds (see Fig. 9), its slow convergence rate will make it suffer from more leakage before reaching a target accuracy rate. For example, given a target accuracy of 85% on the MNIST dataset, both FedAvg and FedProx achieve the target accuracy with 0.058% and 0.057% leakage while FedSGD will reach 85% accuracy in later rounds resulting in an accumulative MI leakage of 0.11% (even with smaller leakage per round).



(a) Normalized MI, MNIST. **(b) Normalized MI, CIFAR10.**

Figure 11: Impact of the local training round (E) when using FedAvg. We normalize the MI by the entropy of each user's local dataset, and we consider $N \in \{10, 20\}$.



(a) Normalized MI, MNIST. **(b) Normalized MI, CIFAR10.**

Figure 12: Impact of the local training round (E) when using FedProx. We normalize the MI by the entropy of each user's local dataset, and we consider $N \in \{10, 20\}$.

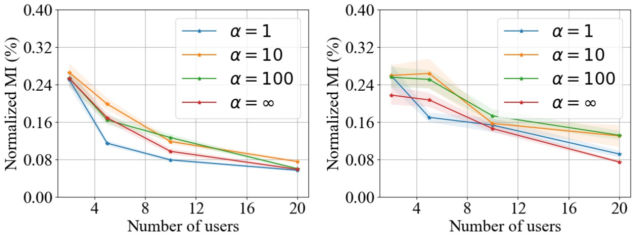
5.5 Impact of Local Training Epochs (E)

Fig. 11 shows the impact of varying the number of local training epochs E on MI privacy leakage in FedAvg on both MNIST and CIFAR10 datasets. We select E from $\{1, 2, 5, 10\}$ and N from $\{10, 20\}$, and we consider MLP model for MNIST and CNN model for CIFAR10. We observe that increasing the local training round E will increase the MI privacy leakage in FedAvg. An intuitive explanation is that with more local epochs, the local model updates become more biased towards the user's local dataset, hence it will potentially leak more private information about users' and make it easier for the server to infer the individual model update from the aggregated update. However, as shown in Fig. 11, increasing the local epochs E will not have a linear impact on the increase of MI privacy leakage. As E increases, the increase rate of MI privacy leakage becomes smaller.

Similar to FedAvg, we observe from Fig. 12 that the local training epochs E has a sub-linear impact on the MI privacy leakage when using FedProx. As aforementioned, this can be attributed to the fact that FedProx represents an application of FedAvg with the original loss function in addition to a convex regularization term.

5.6 Impact of Data Heterogeneity

As discussed in Remark 3 of Section 3, in our theoretical analysis, we considered IID data distribution across users in Theorem 1 in order to make use of entropic central limit theorem results in developing our upper bounds on privacy leakage. However in practice, the data distribution at the users can be heterogeneous.



(a) Normalized MI when $E = 1$. **(b) Normalized MI when $E = 5$.**

Figure 13: Impact of user heterogeneity when using FedAvg on non-IID CIFAR10. Note that $\alpha = \infty$ means that the user data distributions are identical (IID users), and the MI is normalized by the entropy of a user's local dataset.

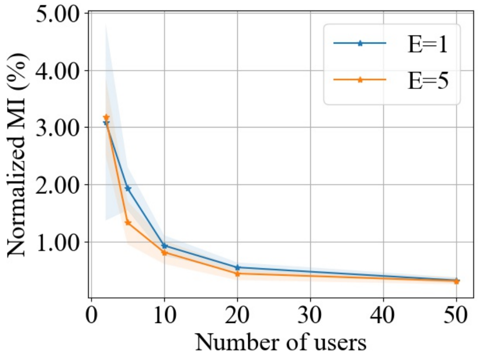


Figure 14: Impact of user heterogeneity when using FedAvg on FEMNIST. Note that the MI is normalized by the entropy of target user's local dataset, which is $678 * 176$.

Hence, in this subsection, we analyze the impact of the non-IID (heterogeneous) data distribution across the users' on the privacy leakage. To measure how user heterogeneity can potentially impact the MI privacy leakage in FedAvg, we consider two different data settings. In the first setting, we create synthetic users with non-IID data distributions following the methodology in [21]. For the second setting, we consider FEMNIST [10], a benchmark non-IID FL dataset extended from MNIST, which consists of 62 different classes of 28×28 images (10 digits, 26 lowercase letters, 26 uppercase letters) written by 3500 users.

In the first, synthetic non-IID data setting, we use Dirichlet distribution parameterized by α to split the dataset into multiple non-IID distributed local datasets. Smaller α (i.e., $\alpha \rightarrow 0$) represents that the users' datasets are more non-identical with each other, while larger α (i.e., $\alpha \rightarrow \infty$) means that the user datasets are more identical with each other. We choose CIFAR10 as the dataset, CNN as the model, and use FedAvg for a case study while using a batch size of $B = 32$. Note that we do not consider FedSGD since it will not be affected by user heterogeneity. During the experiments, we choose the α value from $\{1, 10, 100, \infty\}$ to create different levels of non-IID user datasets, and we consider $N \in \{2, 5, 10, 20\}$ and $E \in \{1, 5\}$.

Fig. 13 shows how the MI privacy leakage varies with the number of users under different α, where the MI privacy leakage is normalized by the entropy of each user's local dataset. We notice that the MI privacy leakage will decrease with the number of users consistently under different α, which empirically shows that our theoretical results in Section 3 also holds in the case where users are heterogeneous.

For the second, FEMNIST data setting, we split the dataset by users into 3500 non-overlapping subsets, each of which contains character images written by a specific user. Considering that the size of each subset is small, in order to have enough training data, we choose to sample N users at each training round instead of using a fixed set of N users, which simulates the user sampling scenario in FL. Specifically, at the beginning of each FL training round with N participating users, we use the same target user and randomly pick the other $N-1$ out of 3500 users. Note that we consider $N \in \{2, 5, 10, 20, 50\}$ and $E \in \{1, 5\}$, and use the same model (CNN), batch size ($B = 32$), and FedAvg algorithm in our evaluation..

Fig. 14 shows how the MI privacy leakage varies with the number of users. Similar to the synthetic non-IID data setting in Fig. 13, the privacy leakage decreases with increasing the number of user N.

5.7 Practical Privacy Implications

Success of Privacy attacks. To provide insights on how MI translates to practical privacy implications, we conduct experiments using one of the state-of-the-art data reconstruction attack, i.e., the Deep Leakage from Gradients (DLG) attack from [44], to show how the MI metric reflects the reconstructed image quality of the attack as we vary system parameters. Specifically, we choose MNIST as the dataset, the same SLP used in Section 4.2 as the model, and FedSGD with batch size of 32 as training algorithm. For the data distribution across the users, we consider the IID setting. At the end of each training round, each user uses a batch of images with size 32 to calculate their local gradients, which will be securely aggregated by the server. The DLG attack will reconstruct a batch of images with size 32 from the aggregated gradient, making them as similar as possible to the batch of images used by the target user. After that, we apply the same PSNR (Peak Signal-to-noise Ratio) metric used in [44] to measure the quality of reconstructed images compared with the images used by the target user during training. Note that without loss of generality, we report the PSNR value of reconstructed images by DLG attack for the first training round.

Fig. 15 shows the impact of number of users N on the privacy leakage metric (MI) and the reconstructed image quality of DLG attack (PSNR). We pick the image of digit 3 out of the target 32 images as an example of reconstructed images. We can observe that increasing the number of users N decreases the MI metric as well as the PSNR at almost the same rate. This demonstrates that the MI metric used in this paper can translate to practical privacy implications well.

MI Privacy leakage under the joint use of DP and SA. To highlight the joint effect of differential privacy with secure aggregation, we conduct experiments on the MNIST dataset with a linear model to measure the MI privacy leakage in the presence of centralized DP noise added at the server after SA. Specifically, following [1], we first clip the aggregated model updates to make its norm bounded by C, and then add Gaussian noise with variance σ^2 to achieve (ϵ, δ)-DP. We set $C = 1$, $\delta = 1/1200$, and $\sigma = \sqrt{2\log(\frac{1.25}{\delta})}/\epsilon$.

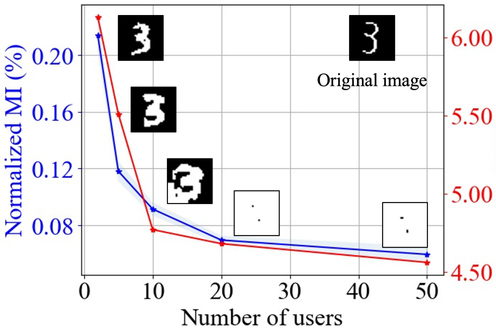


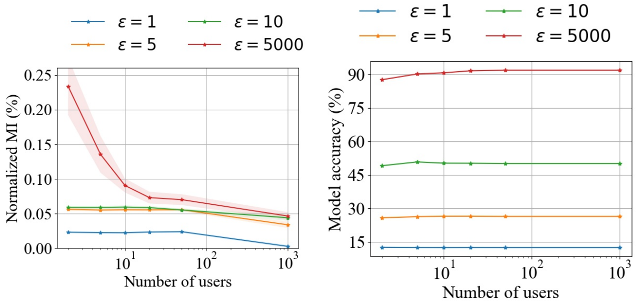
Figure 15: Impact of varying the number of users N, on the reconstructed image quality (PSNR) of the DLG attack and on the MI privacy leakage.

Fig. 16a shows the MI privacy leakage for different (ϵ, δ)-DP levels with SA (δ is fixed at $1/1200$). As the number of users increase, SA improves the privacy level (measured in terms of MI leakage) for different levels of DP noise, with the effect being most pronounced for weak DP noise level ($\epsilon = 5000$ in Fig. 16a). Our experiments also show that as the number of users increase, the gain from using higher DP noise levels is diminished. In particular, with $N = 1000$ users, the MI leakage level for ϵ =5, 10 and 5000 are almost the same; MI leakage is only reduced from 0.046% to 0.034% when using $\epsilon = 5$ instead of $\epsilon = 5000$. In contrast, we get a reduction from 0.234% to 0.056% when there are $N = 2$ users.

Importantly, the reduction observed in privacy leakage due to applying additional DP noise results in a severe degradation in accuracy as seen in Fig. 16b, whereas privacy improvement gained by having more users has a negligible effect on the performance of the trained model. For example, consider the case of 1000 users. One may achieve the same level of privacy in terms of MI leakage (lower than 0.05% MI) with either (i) (ϵ, δ)-DP with $\epsilon = 10$, which, however, results in unusable model accuracy (less than 50%), or, (ii) by aggregating the 1000 users and using a tiny amount of DP noise (equivalent to $\epsilon = 5000$), which achieves a model accuracy higher than 90%.

6 RELATED WORK

Secure Aggregation in FL. As mentioned secure aggregation has been developed for FL [9] to provide protection against model inversion attacks and robustness to user dropouts (due to poor connections or unavailability). There has been a series of works that aim at improving the efficiency of the aggregation protocol [7, 16, 22, 35–37, 43]. This general family of works using secure aggregation disallow the learning information about each client's individual model update beyond the global aggregation of updates, however there has not been a characterization of how much information the global aggregation can leak about the individual client's model and dataset. To the best of our knowledge, in this work, we provide the first characterization of the privacy leakage due to the aggregated model through mutual information for FL using secure aggregation.



(a) Normalized MI, MNIST.

(b) Model accuracy, MNIST.

Figure 16: Effects of using DP noise together with SA on MI privacy leakage and model accuracy. Note that we add DP noise in aggregated model updates after SA.

Differential Privacy. One way to protect a client's contributions is to use differential privacy (DP). DP provides a rigorous, worst-case mathematical guarantee that the contribution a single client does not impact the result of the query. Central application of differential privacy was studied in [1, 5, 11]. This form of central application of DP in FL requires trusting the server with individual model updates before applying the differentially private mechanism. An alternative approach studied in FL for an untrusted server entity is the local differential privacy (LDP) model [2, 4, 25] were clients apply a differentially private mechanism (e.g. using the Gaussian mechanism) locally on their update before sending to the central server. LDP constraints imply central DP constraints, however due to local privacy constraints LDP mechanisms significantly perturb the input and reduces globally utility due to the compounded effect of adding noise at different clients.

In this work, we use a mutual information metric to study the privacy guarantees for the client's dataset provided through the secure aggregation protocol without adding differential privacy noise at the clients. In this case, secure aggregation uses contributions from other clients to mask the contribution of a single client. We will discuss in Section 7 situations where relying only on SA can clearly fail to provide differential privacy guarantees and comment on the prevalence of such situations in practical training scenarios.

Privacy Attacks. There have been some works trying to empirically show that it is possible to recovery some training data from the gradient information. [3, 32, 39, 41]. Recently, the authors in [18] show that it is possible to recover a batch of images that were used in the training of non-smooth deep neural network. In particular, their proposed reconstruction attack was successful in reconstruction of different images from the average gradient computed over a mini-batch of data. Their empirical results have shown that the success rate of the inversion attack decreases with increasing the batch size. Similar observations have been demonstrated in the subsequent works [41]. In contrast to this work, we are the first to the best of our knowledge to theoretically quantify the amount of information that the aggregated gradient could leak about the private training data of the users, and to understand how the training parameters (e.g., number of users) affect the leakage. Additionally, our empirical results are different from the ones in [3, 32, 39, 41, 41] in the way of quantifying the leakage. In particular, we use the MINE tool to abstractly quantify the amount of information leakage in bits instead of the number of the reconstructed images. We have also empirically studied the effect of the system parameters extensively using different real world data sets and different neural network architectures.

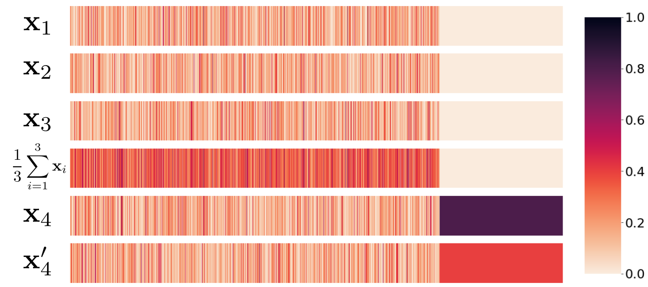


Figure 17: Heatmap of the absolute values of sampled updates from clients 1, 2 and 3 in the counter example. $\mathbf{x}_4$ and $\mathbf{x}_4'$ can be distinguished even adding the aggregated noise from $\sum_{i=1}^{3}\mathbf{x}_i$.

7 FURTHER DISCUSSION AND CONCLUSIONS

In this paper, we derived the first formal privacy guarantees for FL with SA using MI as a metric to measure how much information the aggregated model update can leak about the local dataset of each user. We proved theoretical bounds on the MI privacy leakage in theory and showed through an empirical study that this holds in practice after FL settings. Our concluding observations is that by using FL with SA, we get that: 1) the MI privacy leakage will decrease at a rate of $O(\frac{1}{N})$ (N is the number of users participating in FL with SA); 2) increasing model size will not have a linear impact on the increase of MI privacy leakage, and the MI privacy leakage only linearly increases with the rank of the covariance matrix of the individual model update; 3) larger batch size during local training can help to reduce the MI privacy leakage. We hope that our findings can shed lights on how to select FL system parameters with SA in practice to reduce privacy leakage and provide an understanding for the baseline protection provided by SA in settings where it is combined with other privacy-preserving approaches such as differential privacy.

Can we provide differential privacy guarantees using SA? Note that when using FL with SA, then from the point of view of an adversary that is interested in the data of the i-th user, the aggregated model in $i^- = [N]\backslash\{i\}$ can be viewed as noise that is independent of the gradient $\mathbf{x}_i$ given the last global model, which is very similar to an LDP mechanism for the update $\mathbf{x}_i^{(t)}$ of user i that adds noise to $\mathbf{x}_i^{(t)}$. This leads to an intriguing question: *Can we get LDP-like guarantees from the securely aggregated updates?*

Since DP is interested in a worst-case guarantee, it turns out that their exist model update distributions where it is impossible to achieve an $\epsilon < \infty$ DP guarantee by using other model updates as noise as illustrated in Fig. 17. In this case, the alignment of the sparsity pattern in x_1, x_2 and x_3 allows an adversary to design a perfect detector to distinguish between x_4 and x_4'.

Why our MI privacy guarantee can avoid this? Although, the previous example illustrates that DP flavored guarantees are not always possible, in practical scenarios, the worst-case distribution for $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$ that enables the distinguishing between $\mathbf{x}_4$ and $\mathbf{x}_4'$ in Fig. 17 are an unlikely occurrence during training. For instance, in our theoretical analysis, since users have IID datasets, then having the distribution of x_1, x_2 and x_3 be restricted to a subspace $\mathcal{S}_{x_{i^-}}$, implies also that points generated from $\mathbf{x}_4$ would also belong to $\mathcal{S}_{x_{i^-}}$ almost surely. This is a key reason why we can get mutual information guarantee in Theorem 1: for an aggregated gradient direction $\sum_{i=1}^{N} \mathbf{x}_i$, where each component is restricted to a common subspace $\mathcal{S}_x$ protects the contribution of each individual component $\mathbf{x}_i$ as N increases.

In the worst case, where one component is not restricted to the subspace $\mathcal{S}_x$ spanned by the remaining components, then we get the privacy leakage discussed in the example above. We highlight that through our experiments and other studies in the literature [17], we observe that such sparsity alignment happens with very low probability. This presents motivation for studying a probabilistic notion of DP that satisfies (ϵ, δ)-DP with a probability at least γ, instead of the worst-case treatment in current DP notions, but this is beyond the scope of the study in this current work.

Another interesting future direction is to use the results from this work for a providing "privacy metrics" to users to estimate/quantify their potential leakage for participating in a federated learning cohort. Such metrics can be embedded in platforms, such as FedML [20], to guide users to make informed decisions about their participation in federated learning. Finally, it would also be important to extend the results to model aggregation protocols that are beyond weighted averaging (e.g., in federated knowledge transfer [19]).

ACKNOWLEDGMENTS

This paper is based upon work supported by the National Science Foundation (NSF) under grants CNS-1956435, CNS-1901488, CCF-1703575, CCF-1763673, CNS-2002874, the Defense Advanced Research Projects Agency (DARPA) under Contract No. FASTNICS HR001120C0088, and gifts from Intel/Avast via the PrivateAI institute, Amazon via the USC + Amazon Center on Secure & Trusted ML, Cisco, Konica Minolta, and Qualcomm.

REFERENCES

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

[2] Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. 2018. cpSGD: Communication-efficient and differentially-private distributed SGD. *Advances in Neural Information Processing Systems* 31 (2018).

[3] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. 2017. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security* 13, 5 (2017), 1333–1345.

[4] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The privacy blanket of the shuffle model. In *Annual International Cryptology Conference*. Springer, 638–667.

[5] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE, 464–473.

[6] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual Information Neural Estimation. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 531–540.

[7] James Henry Bell, Kallista A Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. 2020. Secure single-server aggregation with (poly) logarithmic overhead. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1253–1269.

[8] Sergey G Bobkov, Gennadiy P Chistyakov, and Friedrich Götze. 2014. Berry–Esseen bounds in the entropic central limit theorem. *Probability Theory and Related Fields* 159, 3-4 (2014), 435–478.

[9] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1175–1191.

[10] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. 2018. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097* (2018).

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

[12] Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA.

[13] Ye Dong, Xiaojun Chen, Liyan Shen, and Dakui Wang. 2020. EaSTFLy: Efficient and secure ternary federated learning. *Computers & Security* 94 (2020), 101824.

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

[15] Ronen Eldan, Dan Mikulincer, and Alex Zhai. 2020. The CLT in high dimensions: quantitative bounds via martingale embedding. *The Annals of Probability* 48, 5 (2020), 2494–2524.

[16] Ahmed Roushdy Elkordy and A. Salman Avestimehr. 2022. HeteroSAg: Secure Aggregation with Heterogeneous Quantization in Federated Learning. *IEEE Transactions on Communications* (2022), 1–1. https://doi.org/10.1109/TCOMM.2022.3151126

[17] Irem Ergun, Hasin Us Sami, and Basak Guler. 2021. Sparsified Secure Aggregation for Privacy-Preserving Federated Learning. *arXiv preprint arXiv:2112.12872* (2021).

[18] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting Gradients – How easy is it to break privacy in federated learning?. In *Advances in Neural Information Processing Systems*.

[19] Chaoyang He, Murali Annavaram, and Salman Avestimehr. 2020. Group knowledge transfer: Federated learning of large cnns at the edge. *Advances in Neural Information Processing Systems* 33 (2020), 14068–14080.

[20] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, et al. 2020. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518* (2020).

[21] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335* (2019).

[22] Swanand Kadhe, Nived Rajaraman, O Ozan Koyluoglu, and Kannan Ramchandran. 2020. Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. *arXiv preprint arXiv:2009.11248* (2020).

[23] Peter Kairouz, Ziyu Liu, and Thomas Steinke. 2021. The distributed discrete gaussian mechanism for federated learning with secure aggregation. *arXiv preprint arXiv:2102.06387* (2021).

[24] Peter Kairouz, H. Brendan McMahan, Brendan, and et al. 2019. Advances and Open Problems in Federated Learning. *preprint arXiv:1912.04977* (2019). arXiv:1912.04977

[25] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. What can we learn privately? *SIAM J. Comput.* 40, 3 (2011), 793–826.

[26] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images.* Technical Report. Citeseer.

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012).

[28] Eugene Kuznetsov, Yitao Chen, and Ming Zhao. 2021. SecureFL: Privacy Preserving Federated Learning with SGX and TrustZone. In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*. 55–67. https://doi.org/10.1145/3453142.3491287

[29] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[30] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Jerry Zhu (Eds.). 1273–1282.

[31] Vaikkunth Mugunthan, Antigoni Polychroniadou, David Byrd, and Tucker Hybinette Balch. 2019. Smpai: Secure multi-party computation for federated learning. In *Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services*.

[32] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. 2017. Privacy-preserving deep learning: Revisited and enhanced. In *International Conference on Applications and Techniques in Information Security*. Springer, 100–110.

[33] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. 2018. On the convergence of federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127* 3 (2018), 3.

[34] Umut Simsekli, Levent Sagun, and Mert Gurbuzbalaban. 2019. A tail-index analysis of stochastic gradient noise in deep neural networks. In *International Conference on Machine Learning*. PMLR, 5827–5837.

[35] Jinhyun So, Ramy E Ali, Basak Guler, Jiantao Jiao, and Salman Avestimehr. 2021. Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning. *arXiv preprint arXiv:2106.03328* (2021).

[36] Jinhyun So, Başak Güler, and A Salman Avestimehr. 2021. Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning. *IEEE Journal on Selected Areas in Information Theory* 2, 1 (2021), 479–489.

[37] Jinhyun So, Corey J Nolet, Chien-Sheng Yang, Songze Li, Qian Yu, Ramy E Ali, Basak Guler, and Salman Avestimehr. 2022. Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. *Proceedings of Machine Learning and Systems* 4 (2022), 694–720.

[38] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. 2019. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. 1–11.

[39] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2512–2520.

[40] Runhua Xu, Nathalie Baracaldo, Yi Zhou, Ali Anwar, and Heiko Ludwig. 2019. Hybridalpha: An efficient approach for privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. 13–23.

[41] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. 2021. See through Gradients: Image Batch Recovery via GradInversion. *arXiv,2104.07586* (2021).

[42] Yuhui Zhang, Zhiwei Wang, Jiangfeng Cao, Rui Hou, and Dan Meng. 2021. ShuffleFL: gradient-preserving federated learning using trusted execution environment. In *Proceedings of the 18th ACM International Conference on Computing Frontiers*. 161–168.

[43] Yizhou Zhao and Hua Sun. 2021. Information theoretic secure aggregation with user dropouts. In *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 1124–1129.

[44] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep Leakage from Gradients. In *Advances in Neural Information Processing Systems*, Vol. 32.

[45] Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. 2019. The Anisotropic Noise in Stochastic Gradient Descent: Its Behavior of Escaping from Sharp Minima and Regularization Effects. In *International Conference on Machine Learning*. PMLR, 7654–7663.

A PROOF OF THEOREM 1

Without loss of generality, using permutation of clients indices, we will prove the upper bound for the following term

$$I\left(\mathbf{x}_N^{(t)}; \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^{(t)} \middle| \left\{\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^{(k)}\right\}_{k\in[t-1]}\right), \tag{17}$$

where $\mathbf{x}_N$ is the mini-batch gradient of node i which is given by

$$\mathbf{x}_i^{(t)} = \frac{1}{B}\sum_{b\in\mathcal{B}_i^{(t)}} g_i(\boldsymbol{\theta}^{(t)}, b), \tag{18}$$

We will use the following property of vectors with singular covariance matrices in the proof of this theorem.

Property 1. *Given a random vector* $\mathbf{q}$ *with a singular covariance matrix* $\mathbf{K}_q$ *of rank* d^**, there exists a sub-vector* $\bar{\mathbf{q}}$ *of* $\mathbf{q}$ *with a non-singular covariance matrix* $\mathbf{K}_{\bar{q}}$ *such that* $\mathbf{q} = \mathbf{A}\bar{\mathbf{q}}$ *where* $\mathbf{A} \in \mathbb{R}^{d\times d^*}$ *is a deterministic linear transformation matrix.*

Let us define $S_N^{(t)} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^{(t)}$. We also use the definition of $\bar{g}_i(\boldsymbol{\theta}^{(t)}, b) \in \mathbb{R}^{d^*}$, for $d^* \leq d$ where d is the model size, which is the largest sub-vector of the stochastic gradient $g_i(\boldsymbol{\theta}^{(t)}, b)$ such that $\bar{g}_i(\boldsymbol{\theta}^{(t)}, b)$ has a non-singular covariance matrix $K_{\bar{G}^{(t)}}$ for all $i \in \mathcal{N}$. According to the definition of $\bar{g}_i(\boldsymbol{\theta}^{(t)}, b)$, we can rewrite (17) and the term $S_N^{(t)}$ as follows:

$$\begin{aligned}\bar{\mathbf{x}}_i^{(t)} &= \frac{1}{B}\sum_{b\in\mathcal{B}_i^{(t)}}\bar{g}_i(\boldsymbol{\theta}^{(t)}, b)\\ \bar{S}_N^{(t)} &= \frac{1}{N}\sum_{i\in\mathcal{N}}\bar{\mathbf{x}}_i^{(t)}\end{aligned} \tag{19}$$

Let also define $F_N^{(t)} = \sqrt{N}\bar{S}_N^{(t)}$. We can decompose the expression in (17) as follows:

$$\begin{aligned}
&I\left(\mathbf{x}_N^{(t)}; S_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\overset{(a)}{=} I\left(\sqrt{B}\mathbf{x}_N^{(t)}; \sqrt{N}S_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\overset{(b)}{=} I\left(\sqrt{B}\bar{\mathbf{x}}_N^{(t)}; F_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&= h\left(\sqrt{B}\bar{\mathbf{x}}_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) + h\left(F_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\qquad - h\left(\sqrt{B}\bar{\mathbf{x}}_N^{(t)}, F_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&= h\left(\sqrt{B}\bar{\mathbf{x}}_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) + h\left(F_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\qquad - h\left(\begin{bmatrix}\mathrm{I}_d^* & 0_d^*\\ \mathrm{I}_d^*\frac{1}{\sqrt{N}} & \frac{\sqrt{N-1}}{\sqrt{N}}\mathrm{I}_d^*\end{bmatrix}\begin{bmatrix}\sqrt{B}\bar{\mathbf{x}}_N^{(t)}\\ F_{N-1}^{(t)}\end{bmatrix} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\overset{(c)}{=} h\left(\sqrt{B}\bar{\mathbf{x}}_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) + h\left(F_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\qquad - h\left(\sqrt{B}\bar{\mathbf{x}}_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) - h\left(F_{N-1}^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\qquad - \log\left|\det\begin{bmatrix}\mathrm{I}_d^* & 0_d^*\\ \mathrm{I}_d^*\frac{1}{\sqrt{N-1}} & \frac{\sqrt{N-1}}{\sqrt{N}}\mathrm{I}_d^*\end{bmatrix}\right|\\
&\overset{(d)}{=} h\left(F_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\qquad - h\left(F_{N-1}^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right),
\end{aligned} \tag{20}$$

where: (a) follows from the fact that the mutual information is invariant under deterministic multiplication; (b) from Property 1 (c) follows from the property of the entropy of linear transformation of random vectors [12] and the fact that $\bar{\mathbf{x}}_N^{(t)}$ and $F_{N-1}^{(t)}$ are conditionally independent given $\left\{S_N^{(k)}\right\}_{k\in[t-1]}$ (e.g., the last global model at time t) ; (d) follows from the Schur compliment of the matrix.

We will now turn our attention to characterizing the entropy term $h\left(F_M^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)$ for any M. Note that

$$
\begin{aligned}
&h\left(F_M^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&= h\left(\frac{1}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{d\in\mathcal{B}_i^{(t)}}\bar{g}_i(b,\boldsymbol{\theta}^{(t)})\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\overset{(i)}{=} h\left(\frac{\mathbf{K}_{\bar{G}^{(t)}}^{1/2}}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{b\in\mathcal{B}_i^{(t)}}\widehat{g}_i(b,\boldsymbol{\theta}^{(t)})\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\overset{(ii)}{=} \log\left(|\det\mathbf{K}_{\bar{G}^{(t)}}^{1/2}|\right)\\
&\quad+\underbrace{h\left(\frac{1}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{b\in\mathcal{B}_i^{(t)}}\widehat{g}_i(b,\boldsymbol{\theta}^{(t)})\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)}_{H_M}
\end{aligned}
\tag{21}
$$

where: (i) makes use of the fact that the covariance matrix is the same across clients and using the whitening definition (Definition 1) on the vector $\bar{g}_i(b,\boldsymbol{\theta}^{(t)})$; ($ii$) again uses the property of entropy of linear transformation of random vectors.

Note that the term of $h\left(F_M^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)$ only depends on M in the second term H_M. As a result by substituting (21) in (20), we get that

$$
\begin{aligned}
&I\left(\mathbf{x}_N^{(t)};S_N^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&= H_N - H_{N-1} + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right),
\end{aligned}
\tag{22}
$$

Our final step is to find suitable upper and lower bounds for H_M to use in (22). Recall for the following arguments that due to whitening, the vector $\widehat{g}_b^{(t)} = \widehat{g}(b,\boldsymbol{\theta}^{(t)})$ has zero mean and identity covariance.

A.1 Upper bound on H_M

The upper bound is the simplest due to basic entropy properties. In particular, the sum $\frac{1}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{b\in\mathcal{B}_i^{(t)}}\widehat{g}_b^{(t)}$ has zero mean and I_{d^*} covariance. Thus,

$$
\begin{aligned}
H_M &= h\left(\frac{1}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{b\in\mathcal{B}_i^{(t)}}\widehat{g}_b^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&\overset{(a)}{\leq} \frac{1}{2}d^*\log(2\pi e),
\end{aligned}
\tag{23}
$$

where (a) follows from the fact that for a fixed first and second moment, Gaussian distribution maximizes the entropy.

The distinction between the proof of the bound in Case 1 and Case 2 in Theorem 1 is in the lower bound on the term H_M. We start by providing the lower bound that is used for proving Case 1.

A.2 Lower bound on H_M for Case 1 in Theorem 1

For the lower bound, we will rely heavily on the assumption that the elements of $\widehat{g}_b^{(t)}$ are independent and the interesting result that gives Berry-Esseen style bounds for the entropic central limit theorem [8]. In particular, in its simplest form, the result states that for IID zero mean random variables X_i, the entropy of the normalized sum $T_m = \frac{1}{\sqrt{M}}\sum_{i=1}^{M}X_i$ approaches the entropy of a Gaussian random variable Φ_{σ^2} with the same variance σ^2 as X_i, such that the following is always satisfied

$$
h(\Phi_{\sigma^2}) - h(T_M) \leq \tilde{C}\frac{\mathbb{E}|X_i|^4}{M},
\tag{24}
$$

Using (24), we can find a lower bound for H_M as follows:

$$
\begin{aligned}
H_M &= h\left(\frac{1}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{b\in\mathcal{B}_i^{(t)}}\widehat{g}_b^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)\\
&= \sum_{j=1}^{d^*}\underbrace{h\left(\frac{1}{\sqrt{MB}}\sum_{b\in\mathcal{B}_i^{(t)}}\sum_{i=1}^{M}\widehat{g}_b^{(t)}[j]\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)}_{\text{variance = 1}}\\
&\overset{(24)}{\geq}\sum_{j=1}^{d^*}\left(h(\Phi_1) - \frac{C_{0,\bar{g}}}{B}\right) = \frac{d^*}{2}\log(2\pi e) - \frac{C_{0,\bar{g}}}{MB}.
\end{aligned}
\tag{25}
$$

In other words, we have the following bound on H_M

$$
\frac{d^*}{2}\log(2\pi e) - \frac{d^*C_{0,\bar{g}}}{MB} \leq H_M \leq \frac{d^*}{2}\log(2\pi e).
\tag{26}
$$

By substituting (26) in (22) (lower bound for $M = N-1$ and upper bound for $M = N$), we get that

$$
\begin{aligned}
I\left(\mathbf{x}_N^{(t)};S_N^{(t)}\middle|\left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) &= H_N - H_{N-1} + d^*\log\left(\frac{N}{N-1}\right)\\
&\leq \frac{d^*}{2}\log\left(\frac{N}{N-1}\right) + \frac{d^*C_{0,\bar{g}}}{(N-1)B}.
\end{aligned}
\tag{27}
$$

This concludes the proof of Case 1.

A.3 Lower bound on H_M for Case 2 in Theorem 1

The proof of this lower bound relies on the entropic central limit theorem for the vector case [15] and Lemma 1 below. We start by giving the entropic central limit theorem for the case of IID random vectos [15].

Theorem 2 (Entropic central limit theorem [15])**.** *Let* $\mathbf{q}$ *be a* σ*-uniformly log concave* d*-dimensional random vector with* $\mathbb{E}[\mathbf{q}] = 0$ *and non-singular covariance matrix* Σ*. Additionally, let* $\mathbf{z} \sim \mathcal{N}(0,\Sigma)$ *be a Gaussian vector with the same covariance as* $\mathbf{q}$*, and let* $\gamma \sim \mathcal{N}(0,\mathrm{I}_d)$ *to be a standard Gaussian. The entropy of the normalized sum* $T_M = \frac{1}{\sqrt{M}}\sum_{i=1}^{M}\mathbf{q}_i$*, where* $\mathbf{q}_i$*'s are random samples, approaches the entropy of a Gaussian random vector* Z*, such that the following is always satisfied*

$$
Ent(T_M||\mathbf{z}) \leq \frac{2(d + 2(Ent(\sqrt{\sigma}\mathbf{q}||\gamma)}{M\sigma^4},
\tag{28}
$$

where $Ent(T_M||\mathbf{z})$ is the relative entropy.

Lemma 1. *Given a random vector $\mathbf{q} \in \mathbb{R}^d$ with a distribution $f_{\mathbf{q}}(y)$ and $Cov(\mathbf{q}) = \Sigma$, and defining $\mathbf{z} \sim \mathcal{N}(0, \Sigma)$ to be a Gaussian vector with the same covariance as $\mathbf{q}$, for $\sigma > 0$, we get*

$$Ent(\sqrt{\sigma}\mathbf{q}||\mathbf{z}) = -h(\mathbf{q}) - \frac{d}{2}\log(\sigma) + \frac{d}{2}\log(2\pi) + \frac{1}{2}\log(|\Sigma|) + \sigma\frac{d}{2}, \tag{29}$$

and

$$Ent(\mathbf{q}||\mathbf{z}) = h(\mathbf{z}) - h(\mathbf{q}). \tag{30}$$

Given the assumption that $\widehat{g}_b^{(t)}$ has a σ-log concave distribution while both the term $\frac{1}{\sqrt{MB}}\sum_{i=1}^{M}\sum_{b\in\mathcal{B}_i^{(t)}}\widehat{g}_b^{(t)}$ and $\widehat{g}_b^{(t)}$ have an identity covariance matrix $\Sigma = \mathrm{I}_{d^*}$ given $\left\{S_N^{(k)}\right\}_{k\in[t-1]}$, we can use (28) with $\mathbf{z} \sim \mathcal{N}(0, \mathrm{I}_{d^*})$. Furthermore, by using Lemma 1, we get

$$h(\mathbf{z}) - H_M \leq \frac{d^* C_{1,\bar{g}} - C_{2,\bar{g}}}{MB}, \tag{31}$$

where, $C_{1,\bar{g}} = \frac{2(1+\sigma+\log(2\pi)-\log(\sigma))}{\sigma^4}$ and $C_{2,\bar{g}} = \frac{4h\left(\widehat{g}(b,\boldsymbol{\theta}^{(t)})\right)}{\sigma^4}$, and $h(\widehat{g}(b, \boldsymbol{\theta}^{(t)}))$ is the entropy of the random vector $\bar{g}_i(b, \boldsymbol{\theta}^{(t)})$ after whitening.

Finally, using the fact that the entropy of the Gaussian random vector $\mathbf{z}$ with covariance I_{d^*} is given by $h(\mathbf{z}) = \frac{d^*}{2}\log(2\pi e)$, we get the following bound on H_M

$$\frac{d^*}{2}\log(2\pi e) - \frac{d^* C_{1,\bar{g}} - C_{2,\bar{g}}}{(N-1)B} \leq H_M \leq \frac{d^*}{2}\log(2\pi e). \tag{32}$$

By substituting (32) in (22) (lower bound for $M = N-1$ and upper bound for $M = N$), we can now upper bound the mutual information term as follows

$$\begin{aligned} &I\left(\mathbf{x}_N^{(t)}; S_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) \\ &= H_N - H_{N-1} + \frac{d^*}{2}\log\left(\frac{N}{N-1}\right) \\ &\leq \frac{d^*}{2}\log\left(\frac{N}{N-1}\right) + \frac{d^* C_{1,\bar{g}} - C_{2,\bar{g}}}{(N-1)B}. \end{aligned} \tag{33}$$

This concludes the proof of Theorem 1.

B PROOF OF COROLLARY 1

In the following, we define $S_N^{(t)} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^{(t)}$. Using this notation, we can upper bound $I_{\text{priv/data}}$ as follows

$$\begin{aligned} I_{\text{priv/data}} &= I\left(\mathcal{D}_i; \left\{S_N^{(k)}\right\}_{k\in[T]}\right) \\ &\overset{(a)}{=} \sum_{t=1}^{T} I\left(\mathcal{D}_i; S_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) \\ &\overset{(b)}{\leq} \sum_{t=1}^{T} I\left(\mathcal{B}_i^{(t)}; S_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right) \\ &\overset{(c)}{\leq} \sum_{t=1}^{T} \underbrace{I\left(\mathbf{x}_i^{(t)}\left(\mathcal{B}_i^{(t)}; \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right); S_N^{(t)} \middle| \left\{S_N^{(k)}\right\}_{k\in[t-1]}\right)}_{\text{This is bounded by the result in Theorem 1}}. \end{aligned} \tag{34}$$

where: (a) comes from the chain-rule; (b) from data processing inequality $D_i \rightarrow B_i^{(t)} \rightarrow \mathbf{x}_i^{(t)}$, where $B_i^{(t)}$ is the sampled mini-batch from the data set of node i; (c) from data processing inequality $B_i^{(t)} \rightarrow \mathbf{x}_i^{(t)} \rightarrow \frac{1}{N}\sum_{i\in\mathcal{N}}\mathbf{x}_i^{(t)}$;. Combining the results given in the two cases of Theorem 1 with (34) concludes the proof of Corollary 1.

C PROOF OF LEMMA 1

$$\begin{aligned} \mathrm{Ent}(\sqrt{\sigma}\mathbf{q}||\mathbf{Z}) &= \mathrm{Ent}(\mathbf{q}'||\mathbf{Z}) = \int f_{\mathbf{q}'}(y)\log\frac{f_{\mathbf{q}'}(y)}{f_{\mathbf{Z}}(y)}dy \\ &= \int f_{\mathbf{q}'}(y)\log f_{\mathbf{q}'}dy - \int f_{\mathbf{q}'}(y)\log f_{\mathbf{Z}}(y)dy \\ &\overset{(a)}{=} -h(\mathbf{q}') + \frac{d}{2}\log(2\pi) \\ &\quad + \frac{1}{2}\log(|\Sigma|) + \frac{1}{2}\int f_{\mathbf{q}'}(y)y^T\Sigma^{-1}ydy \\ &\overset{(b)}{=} -h(\mathbf{q}) - \frac{d}{2}\log(\sigma) + \frac{d}{2}\log(2\pi) \\ &\quad + \frac{1}{2}\log(|\Sigma|) + \frac{1}{2}\int f_{\mathbf{q}'}(y)\ \mathrm{Tr}(\Sigma^{-1}y^Ty)dy \\ &\overset{(c)}{=} -h(\mathbf{q}) - \frac{d}{2}\log(\sigma) + \frac{d}{2}\log(2\pi) \\ &\quad + \frac{1}{2}\log(|\Sigma|) + \frac{1}{2}\ \mathrm{Tr}\left(\Sigma^{-1}\int f_{\mathbf{q}'}(y)y^Tydy\right) \\ &= -h(\mathbf{q}) - \frac{d}{2}\log(\sigma) + \frac{d}{2}\log(2\pi) \\ &\quad + \frac{1}{2}\log(|\Sigma|) + \frac{1}{2}\ \mathrm{Tr}\left(\Sigma^{-1}\mathbb{E}_{\mathbf{q}'}[\mathbf{q}'^T\mathbf{q}']\right) \\ &\overset{(d)}{=} -h(\mathbf{q}) + \frac{d}{2}\log(\frac{2\pi}{\sigma}) + \frac{1}{2}\log(|\Sigma|) + \frac{1}{2}\sigma\ \mathrm{Tr}\left(\Sigma^{-1}\Sigma\right) \\ &= -h(\mathbf{q}) + \frac{d}{2}\log(\frac{2\pi}{\sigma}) + \frac{1}{2}\log(|\Sigma|) + \sigma\frac{d}{2}, \end{aligned} \tag{35}$$

where: Tr represents the trace function; (a) follows from using the multivariate distribution of the Gaussian vector $\mathbf{z}$; (b) using the scaling property of the entropy with $\mathbf{q}' = \sqrt{\sigma}\mathbf{q}$; (c) from follows from using the linearity of the trace function; finally (d) from using the linear transformation of the random vector $\mathbf{q}' = \sqrt{\sigma}\mathbf{q}$ and the fact that $\mathbf{q}$ has the same covariance matrix Σ as $\mathbf{z}$.

The proof of (30) follows directly by substituting $\sigma = 1$ in the equation (35) and using entropy of a Gaussian vector with covariance Σ.

D OVERVIEW OF MINE

In our empirical evaluation in Section 5, we use the Mutual Information Neural Estimator (MINE) [6] to estimate the mutual information, which is the state-of-the-art method for mutual information estimation [6]. Specifically, given random vectors X and Z, and a function family parameterized by a neural network $\mathcal{F} = \{T_\theta : X \times Z \rightarrow \mathbb{R}\}_{\theta\in\Theta}$, the following bound holds:

$$I(X;Z) \geq I_\Theta(X;Z), \tag{36}$$

where $I_\Theta(X;Z)$ is the neural mutual information measure defined as:

$$I_\Theta(X;Z) = \sup_{\theta\in\Theta}\mathbb{E}_{\mathbb{P}_{XZ}}[T_\theta] - \log(\mathbb{E}_{\mathbb{P}_X\otimes\mathbb{P}_Z}[e^{T_\theta}]), \tag{37}$$

$\mathbb{P}_X$ and $\mathbb{P}_Z$ are the marginal distribution of X and Z respectively, $\mathbb{P}_{XZ}$ is the joint distribution of X and Z, and $\mathbb{P}_X \otimes \mathbb{P}_Z$ is the product of marginals $\mathbb{P}_X$ and $\mathbb{P}_Z$. As an empirical estimation of $I_\Theta(X;Z)$, MINE is implemented as

$$\widehat{I(X;Z)}_K = \sup_{\theta\in\Theta} \mathbb{E}_{\mathbb{P}^{(K)}_{XZ}}[T_\theta] - \log(\mathbb{E}_{\mathbb{P}^{(K)}_X \otimes \mathbb{P}^{(K)}_Z}[e^{T_\theta}]), \tag{38}$$

where $\mathbb{P}^{(K)}_{(\cdot)}$ is the empirical distribution of $\mathbb{P}_{(\cdot)}$ with K IID samples. Finally, solving Eq. 38 (i.e. get the MI estimation) can be achieved by solving the following optimization problem via gradient ascent:

$$\widehat{I(X;Z)}_K = \max_{\theta\in\Theta} \left\{ \frac{1}{K}\sum_{k=1}^{K} T_\theta(x_k, z_k) - \log\left(\frac{1}{K}\sum_{k=1}^{K} e^{T_\theta(x_k,\bar{z}_k)}\right)\right\},$$

where (x_k, z_k) is the k-th sample from $\mathbb{P}_{XZ}$ and $\bar{z}_k$ is the k-th sample from $\mathbb{P}_Z$.