Agent-Level Differentially Private Federated Learning via Compressed Model Perturbation

Yuanxiong Guo*, Rui Hu†, and Yanmin Gong‡
* Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249
† Department of Computer Science and Engineering, University of Nevada, Reno, Reno, NV, 89557
‡ Department of Electrical and Computer Engineering, University of Texas at San Antonio, San Antonio, TX 78249
yuanxiong.guo@utsa.edu, ruihu@unr.edu, yanmin.gong@utsa.edu

***Abstract*—Federated learning (FL) involves a network of distributed agents that collaboratively learn a common model without sharing their raw data. Privacy and communication are two critical concerns of FL, but they are often treated separately in the literature. While random noise can be added during the FL process to defend against privacy inference attacks, its magnitude is linearly proportional to the model size, which can be very large for modern deep neural networks and lead to severe degradation in model accuracy. On the other hand, various compression techniques have been proposed to improve the communication efficiency of federated learning, but their interplay with privacy protection is largely ignored. Motivated by the observation that privacy protection and communication reduction are closely related in the context of FL, we propose a new federated learning scheme called CMP-Fed that achieves agent-level differential privacy with high model accuracy by leveraging the communication compression techniques in FL with large model sizes. The key component of CMP-Fed is compressed model perturbation (CMP), which first compresses the shared model updates before perturbing them with random noise at each communication round of federated learning. Experimental results based on Fashion-MNIST dataset show that CMP-Fed can largely outperform the existing differentially private federated learning schemes in terms of model accuracy under the same privacy guarantee while still enjoying the communication benefit of model compression.**

***Index Terms*—federated learning, differential privacy, model compression, low-rank approximation.**

I. Introduction

Federated learning (FL) enables a network of distributed agents to collaboratively learn a common machine learning model under the orchestration of a central server. Compared with the traditional, centralized learning, FL is capable of reducing communication cost, improving latency, and enhancing data privacy while obtaining an accurate shared learning model for on-device inference [1]. Therefore, FL has attracted significant attention over the past few years and been applied to a wide range of applications such as Google and Apple's smartphone apps, autonomous driving, financial risk prediction, pharmaceuticals discovery, electronic health record mining, and smart manufacturing [2].

Although promising, FL faces several challenges, among which privacy is a major one. Although personal data are kept on agents locally in FL, it can still be inferred from local model updates communicated between agents and the central server as demonstrated by recent attacks including model inversion attack [3] and membership inference attack [4]. As a cryptography-inspired rigorous definition of privacy, differential privacy (DP) has become the de facto standard for achieving data privacy and can defend against various privacy inference attacks in FL [5]. The main challenge to achieve DP in FL lies in the adverse impact of DP noise on model accuracy. Although DP can be straightforwardly achieved using Gaussian or Laplacian mechanism in FL, the required intensity of added noise to achieve a desirable level of DP is high, particularly for deep neural networks (DNNs) with large numbers of model parameters.

Besides privacy, communication is another core challenge in FL as local model updates of large size need to be exchanged frequently, and the network bandwidth between the central server and agents is often limited. In light of that, various communication compression techniques [6]–[9] have been proposed in literature to reduce the size of messages transmitted at each communication round. However, the interplay between communication compression and privacy protection is seldom considered in the literature. Note that communication reduction and privacy protection are actually closely related in the context of FL, and they share a common goal to reduce, mask, or transform information sent across the network in a way that preserves the underlying learning task.

The joint consideration of communication efficiency and privacy protection in FL faces the following major challenges. First, both communication compression techniques (e.g., quantization and sparsification) and DP techniques (e.g., Laplace and Gaussian mechanisms) are known to affect the model accuracy adversely. Combining those techniques in a straightforward manner could lead to severe degradation in model accuracy, making the learned model useless. Second, secure aggregation has become a common practice in FL as a key privacy-enhancing technique and enables the server to obtain the aggregated model update information without knowing individual values from agents [10]–[12]. The existing compression techniques such as stochastic quantization [8] and random/top-k sparsification [13] are not directly compatible with secure aggregation because the compressed updates shared by the agents cannot be aggregated at the server without being decompressed first.

Most of the works in FL literature treat communication efficiency and privacy separately. For a few studies that do consider both, they either simply combine the techniques for addressing each individual goal together without considering their interplay [14], or view those two goals as conflicting and aim to tackle the new challenges to privacy brought by the communication-compression techniques [15], [16]. Therefore, the resulting model accuracy is rather low for differentially private and communication-efficient FL, particularly for DNNs with large numbers of model parameters.

In this paper, we show that well-crafted communication compression techniques can help with privacy in FL, and if integrated with DP mechanisms, lead to privacy amplification effects and improved model accuracy. In particular, we propose a new DP technique called compressed model perturbation (CMP) and use it to design CMP-Fed, a novel differentially private FL scheme that can achieve agent-level DP guarantee while maintaining high model accuracy. CMP is inspired by the low-rankedness of gradients in DNNs as observed in [17], [18] and works by first reducing the dimension of a model update via low-rank approximation and then adding Gaussian noise to the low-dimension counterparts. The proposed CMP ensures the signal-to-noise ratio in the reconstructed noisy model update is larger than that obtained by adding noise directly to the original model update, and hence leads to a higher model accuracy under the same DP guarantee. Moreover, CMP is a linear operation and easy to be combined with secure aggregation without losing the communication benefit of update compression.

In summary, the main contributions of this paper are as follows.

- We propose a novel FL scheme called CMP-Fed that achieves agent-level DP with high model accuracy for FL with large DNNs. CMP-Fed can obtain an estimator of target private model update in each communication round with much lower perturbation variance than traditional model update perturbation under the same level of DP guarantee, leading to higher model accuracy.
- We show that the designed compression method in CMP-Fed is linear in the sense that the server can directly aggregate the noisy compressed model updates without the need to decompress individual updates. As a result, it can be easily combined with secure aggregation, which is a key privacy-enhancing technique to achieve agent-level DP in FL, while preserving the benefit of reduced communication.
- We provide a tight end-to-end privacy analysis of CMP-Fed using Rényi differential privacy (RDP) and evaluate the performance of the proposed CMP-Fed on Fashion-MNIST dataset under both IID and non-IID data distributions. The empirical results demonstrate that our solution can achieve much higher model accuracy than existing methods.

The rest of the paper is organized as follows. Preliminaries on DP and FL are described in Section II. Section III introduces the threat model and presents the proposed CMP-Fed scheme. The experimental evaluation is given in Section IV, Section V reviews the related work, and Section VI concludes the paper.

II. Preliminaries

A. FL Basics

A typical FL system consists of N agents and a central server. Each agent has a local dataset, and all agents collaboratively train a global model $\boldsymbol{x}$ on the collection of their local datasets under the orchestration of the central server. The agents in FL aim to find the optimal global model $\boldsymbol{x}$ by solving the following empirical risk minimization problem while keeping their data locally:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) := \frac{1}{N}\sum_{i=1}^{N} f_i(\boldsymbol{x}), \tag{1}$$

where $f_i(\boldsymbol{x}) = \mathbb{E}_{z\in D_i}[l_i(\boldsymbol{x}; z)]$ represents the local loss function of i-th agent (possibly non-convex), D_i is the local dataset of i-th agent, and z represents a data point sampled from D_i. One of the key features in FL is that data distributions at different agents may be non-IID.

Federated Averaging [1] (FedAvg) is the most widely-used optimization algorithm to solve Problem (1) in the FL setting. It is an iterative process that involves multiple communication rounds. At each communication round,

1) the server randomly selects a subset of agents and distributes the current global model to them,
2) each selected agent runs multiple steps of stochastic gradient descent (SGD) in parallel to update the received global model and sends the model update to the server,
3) the server aggregates all the local model updates to update the global model for the next round of training.

While data locality helps privacy protection, secure aggregation is often stacked with FedAvg to further protect the privacy of individual model updates [10]. Specifically, under the secure aggregation protocol, the agent encrypts the local model update before sending it out, and then the server decrypts the sum of these encrypted messages and recovers the sum of local model updates to update the global model.

B. Differential Privacy

DP has been proposed as a rigorous privacy notion for measuring privacy risk. The classic notion of DP, (ϵ, δ)-DP, is defined as follows [5]:

Definition 1 ((ϵ,δ)-DP)**.** *Given privacy parameters $\epsilon > 0$ and $0 \le \delta \le 1$, a randomized mechanism $\mathcal{M}$ satisfies (ϵ,δ)-DP if for any two neighboring datasets D, D' and any subset of outputs $O \subseteq range(\mathcal{M})$,*

$$\Pr[\mathcal{M}(D) \in O] \le e^{\epsilon} \Pr[\mathcal{M}(D') \in O] + \delta. \tag{2}$$

As a relaxed version of (ϵ,δ)-DP, RDP has been proposed in [19] as follows:

Definition 2 ($(\alpha,\rho(\alpha))$-RDP [19])**.** *Given a real number $\alpha > 1$ and privacy parameter $\rho \ge 0$, a randomized mechanism $\mathcal{M}$*

satisfies (α, ρ)-RDP if for any two adjacent datasets D, D', the Rényi α-divergence between $\mathcal{M}(D)$ and $\mathcal{M}(D')$ satisfies

$$D_\alpha[\mathcal{M}(D) \| \mathcal{M}(D')] := \frac{1}{\alpha - 1} \log \mathbb{E}\left[\left(\frac{\mathcal{M}(D)}{\mathcal{M}(D')}\right)^\alpha\right] \leq \rho(\alpha).$$

Compared to (ϵ, δ)-DP, RDP has a tighter composition bound and thus is more suitable to analyze the end-to-end privacy loss of iterative algorithms. One can convert RDP to (ϵ, δ)-DP for any $\delta > 0$ as follows:

Lemma 1 (From RDP to (ϵ, δ)-DP [20])**.** *If the randomized mechanism $\mathcal{M}$ satisfies $(\alpha, \rho(\alpha))$-RDP, then it also satisfies $(\rho(\alpha) + \frac{\log(1/\delta)}{\alpha - 1}, \delta)$-DP.*

In the following, we provide some useful definitions and conclusions about DP and RDP that will support our main results in this paper.

Definition 3 (l_2-sensitivity [5])**.** *Let $h : \mathcal{D} \to \mathbb{R}^d$ be a query function over a dataset. The l_2-sensitivity of h is defined as $\psi(h) := \sup_{D,D'} \|h(D) - h(D')\|_2$ for any two neighboring datasets D and D'.*

Lemma 2 (Gaussian Mechanism [19])**.** *The Gaussian mechanism $\mathcal{M} = h(D) + Z$ with $Z \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$ satisfies $(\alpha, \alpha\psi^2(h)/2\sigma^2)$-RDP.*

Lemma 3 (RDP for Subsampling Mechanism [20], [21])**.** *For a Gaussian mechanism $\mathcal{M}$ and any m-datapoints dataset D, define $\mathcal{M} \circ$ SUBSAMPLE as 1) subsample without replacement B datapoints from the dataset (denote $q := B/m$ as the sampling ratio); and 2) apply $\mathcal{M}$ on the subsampled dataset as input. Then if $\mathcal{M}$ satisfies $(\alpha, \rho(\alpha))$-RDP with respect to the subsampled dataset for all integers $\alpha \geq 2$, then the new randomized mechanism $\mathcal{M} \circ$ SUBSAMPLE satisfies $(\alpha, \rho'(\alpha))$-RDP with respect to D, where*

$$\rho'(\alpha) \leq \frac{1}{\alpha - 1} \log\left(1 + q^2 \binom{\alpha}{2} \min\{4(e^{\rho(2)} - 1), 2e^{\rho(2)}\} + \sum_{j=3}^{\alpha} q^j \binom{\alpha}{j} 2e^{(j-1)\rho(j)}\right).$$

If ${\sigma'}^2 := \sigma^2/\psi^2(h) \geq 0.7$ and $\alpha \leq (2/3)\sigma^2 \log(1/q\alpha(1 + {\sigma'}^2)) + 1$, $\mathcal{M} \circ$ SUBSAMPLE satisfies $(\alpha, 3.5q^2\psi^2(h)\alpha/\sigma^2)$-RDP.

Lemma 4 (RDP Composition [19])**.** *For randomized mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$ applied on dataset D, if $\mathcal{M}_1$ satisfies (α, ρ_1)-RDP and $\mathcal{M}_2$ satisfies (α, ρ_2)-RDP, then their composition $\mathcal{M}_1 \circ \mathcal{M}_2$ satisfies $(\alpha, \rho_1 + \rho_2)$-RDP.*

C. Agent-Level DP in FL

In FL, the model updates of each participating agent are transmitted separately and thus need to be protected from the untrusted cloud server. Due to the distributed nature of FL, we consider the notion of *agent-level DP* in this work and define the neighboring datasets as follows: two datasets D and D' are neighboring datasets if $D \cup \{D_c\}$ or $D \setminus \{D_c\}$ is identical to D' for an agent c, where D_c denotes all the data points associated with agent c. Therefore, the privacy guarantee holds w.r.t. all data points belonging to that agent. This is stronger than the commonly-used notion of *record-level DP* where only the addition or removal of one data point of an agent is protected, and considered to be more suitable for FL settings with large numbers of agents [22].

As a privacy-enhancing technique, secure aggregation is a common practice in the literature to enable agent-level DP in FL. Secure aggregation is a lightweight instance of cryptographic secure multi-party computation that prevents the server from inspecting individual model updates of agents in FL [10], [11]. It allows the server to learn just an aggregate function of the agents' local model updates, typically the sum, and nothing else, and hence improves privacy. In the typical setting of FL with a single server, secure aggregation is achieved by adding random mask vectors sampled over a finite group on local model updates before sending them out. Specifically, agents generate randomly sampled zero-sum mask vectors locally by working in the space of integers modulo m and sampling the elements of the mask uniformly from $\mathbb{Z}_m$. Secure aggregation in FL ensures that the masked local model update is indistinguishable from random values, revealing no further information to potential adversaries. However, when the server computes the modular sum of all the masked updates, the masks cancel out and the server obtains the exact sum of local model updates. As with the existing works in FL [12], [23]–[25], we ignore the finite precision and modular summation arithmetic associated with secure aggregation in this paper, noting that one can follow the strategy in [26] to transform the real-valued vectors into integers for minimizing the approximation error of recovering the sum.

III. CMP-Fed: Differentially Private FL with Compressed Model Perturbation

In this section, we present CMP-Fed with the goal of preserving high model accuracy under agent-level DP guarantee in FL. We first describe the threat model considered in this work, and then we propose our CMP method against this threat and analyze the privacy guarantee of CMP-Fed.

A. Threat Model

The adversary considered here can be the "honest-but-curious" server or agents in the system. The adversary will honestly follow the designed training protocol but is curious about a target agent's private data and wants to infer it from the shared messages. Furthermore, some agents can collude with the server or each other to infer private information about a specific victim agent. Besides, the adversary could also be the passive outside attacker. These attackers can eavesdrop the shared messages in the execution of the training protocol but will not actively inject false messages into or interrupt message transmissions.

Algorithm 1 CMP-Fed: Federated Learning with Compressed Model Perturbation

```
Require: number of selected agents per round $S$, agent-side and server-side learning rates $\eta$ and $\gamma$, number of communication rounds $T$, local iteration period $K$, clipping thresholds $C_1$ and $C_2$, noise magnitudes $\sigma_1, \sigma_2$, and low-rank approximation coefficient $r$.

Server executes:
1: Initialize $\boldsymbol{x}_0 \in \mathbb{R}^{m\times n}, \boldsymbol{V}_0 \in \mathbb{R}^{n\times r}$
2: for each round $t = 0$ to $T-1$ do
3:     $\mathcal{S}_t \leftarrow$ set of $S$ agents sampled uniformly at random without replacement
4:     Broadcast $\boldsymbol{x}_t$ to all agents in $\mathcal{S}_t$
5:     for each agent $i \in \mathcal{S}_t$ in parallel do
6:         $\mathcal{U}^i_{t+1} \leftarrow$ SubspaceIterate$(i, \boldsymbol{x}_t, \boldsymbol{V}_t)$
7:     end for
8:     Decrypt $\sum_{i\in\mathcal{S}_t} \mathcal{U}^i_{t+1}$ to obtain $\sum_{i\in\mathcal{S}_t} \tilde{\boldsymbol{U}}^i_{t+1}$
9:     $\boldsymbol{U}_{t+1} \leftarrow \frac{1}{S}\sum_{i\in\mathcal{S}_t} \tilde{\boldsymbol{U}}^i_{t+1}$
10:    $\hat{\boldsymbol{U}}_{t+1} \leftarrow$ Orthogonalize$(\boldsymbol{U}_{t+1})$
11:    for each agent $i \in \mathcal{S}_t$ in parallel do
12:        $\mathcal{V}^i_{t+1} \leftarrow$ AgentUpdate$(i, \hat{\boldsymbol{U}}_{t+1})$
13:    end for
14:    Decrypt $\sum_{i\in\mathcal{S}_t} \mathcal{V}^i_{t+1}$ to obtain $\sum_{i\in\mathcal{S}_t} \tilde{\boldsymbol{V}}^i_{t+1}$
15:    $\boldsymbol{V}_{t+1} \leftarrow \frac{1}{S}\sum_{i\in\mathcal{S}_t} \tilde{\boldsymbol{V}}^i_{t+1}$
16:    $\boldsymbol{\Delta}_t \leftarrow \hat{\boldsymbol{U}}_{t+1}\boldsymbol{V}^{\mathsf{T}}_{t+1}$
17:    $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t + \gamma\boldsymbol{\Delta}_t$
18: end for
19: return $\boldsymbol{x}_T$

SubspaceIterate$(i, \boldsymbol{x}_t, \boldsymbol{V}_t)$:
20: $\boldsymbol{x}^i_{t,0} \leftarrow \boldsymbol{x}_t$
21: for $k = 0, \ldots, K-1$ do
22:     Compute an unbiased estimate $\boldsymbol{g}^i_{t,k}$ of $\nabla f_i(\boldsymbol{x}^i_{t,k})$ using local dataset $D_i$
23:     $\boldsymbol{x}^i_{t,k+1} \leftarrow \boldsymbol{x}^i_{t,k} - \eta\boldsymbol{g}^i_{t,k}$
24: end for
25: $\boldsymbol{\Delta}^i_t \leftarrow \boldsymbol{x}^i_{t,K} - \boldsymbol{x}_t$
26: $\boldsymbol{U}^i_{t+1} \leftarrow \boldsymbol{\Delta}^i_t\boldsymbol{V}_t$
27: $\bar{\boldsymbol{U}}^i_{t+1} \leftarrow \boldsymbol{U}^i_{t+1} \times \min\left\{1, \frac{C_1}{\|\boldsymbol{U}^i_{t+1}\|_2}\right\}$
28: $\tilde{\boldsymbol{U}}^i_{t+1} \leftarrow \bar{\boldsymbol{U}}^i_{t+1} + \mathcal{MN}_{m\times r}(0, \mathbb{I}, \sigma_1^2\mathbb{I})$
29: Encrypt $\tilde{\boldsymbol{U}}^i_{t+1}$ and send it to server via secure aggregation

AgentUpdate$(i, \hat{\boldsymbol{U}}_{t+1})$:
30: $\boldsymbol{V}^i_{t+1} \leftarrow (\boldsymbol{\Delta}^i_t)^{\mathsf{T}}\hat{\boldsymbol{U}}_{t+1}$
31: $\bar{\boldsymbol{V}}^i_{t+1} \leftarrow \boldsymbol{V}^i_{t+1} \times \min\left\{1, \frac{C_2}{\|\boldsymbol{V}^i_{t+1}\|_2}\right\}$
32: $\tilde{\boldsymbol{V}}^i_{t+1} \leftarrow \bar{\boldsymbol{V}}^i_{t+1} + \mathcal{MN}_{n\times r}(0, \mathbb{I}, \sigma_2^2\mathbb{I})$
33: Encrypt $\tilde{\boldsymbol{V}}^i_{t+1}$ and send it to the server via secure aggregation
```

B. CMP: Compressed Model Perturbation

The basic learning process of our proposed CMP-Fed scheme follows the overall procedures of FedAvg with several key modifications for privacy protection. Specifically, at each communication round of FL, after computing the local update, each agent uses CMP to first reduce the dimension of its model update via low-rank approximation and then add Gaussian noise to the low-dimensional counterparts of model update before sending them to the server via a secure aggregation protocol (refer to [10], [11]). Next, the server reconstructs the noisy aggregated model update from all participating agents without knowing their individual values and updates the global model for the next round. Note that the use of secure aggregation is a common practice in literature to achieve client-level DP in FL [10], [27], and the design of a new secure aggregation protocol is out the scope of this paper. The entire process of CMP-Fed is summarized in Algorithm 1 and described in detail below.

At the t-th communication round, the server samples a set of S agents (denoted by $\mathcal{S}_t$) uniformly at random without replacement to participate in the training. Agents selected at current round perform two procedures–**SubspaceIterate** (line 6) and **AgentUpdate** (line 12)–to provide their model updates with DP guarantee. Information shared by agents would go through a secure aggregation protocol, where each agent performs an encryption step (line 29 and line 33), and the server performs aggregation and decryption (line 8 and line 14). The server then reconstructs the model update $\boldsymbol{\Delta}_t$ (line 16) and updates the global model $\boldsymbol{x}_t$ (line 17).

In the **SubspaceIterate** procedure, each agent $i \in \mathcal{S}_t$ receives the global model $\boldsymbol{x}_t$ from the server and performs K iterations of SGD (lines 21-24) to obtain a local model update $\boldsymbol{\Delta}^i_t$ (line 25). This model update needs to be perturbed before sharing with the server. As explained before, the intensity of the injected Gaussian noises is linearly proportional to the dimension of the shared model update. To reduce the noise intensity, we use the low-rank approximation to preserve the most important information in the model update while aggressively reducing its size before perturbation.

As with existing work on communication-efficient distributed learning with low-rank approximation [6], [9], we represent the local model update $\boldsymbol{\Delta}^i_t$ as a set of weight matrices, each corresponding to a layer. Note that the parameters of fully-connected layers and their gradients have an inherent matrix structure, and the parameters of convolutional layers can also be interpreted as fully-connected layers applied repeatedly over a 2-dimensional grid of inputs. In the experiments, we approximate each layer independently for better computation efficiency.

For presentation simplicity, we use $\boldsymbol{M}_{m\times n}$ to denote the model update matrix of a single layer that we want to approximate for an agent. The goal of the low-rank approximation is to find two matrices $\boldsymbol{U}_{m\times r}$ and $\boldsymbol{V}_{n\times r}$ such that $||\boldsymbol{M} - \boldsymbol{U}\boldsymbol{V}^T||_2$

is small. Here $\boldsymbol{M}$ is an m by n matrix, subspace $\boldsymbol{U}$ is an m by r matrix, model embedding $\boldsymbol{V}$ is an n by r matrix, and $r \ll \min(m, n)$. The factorization of $\boldsymbol{M}$ can be done via a full singular value decomposition (SVD) at every communication round. After the agent obtains $\boldsymbol{U}$ and $\boldsymbol{V}$ for its model update, it can use $\boldsymbol{U}$ and $\boldsymbol{V}$ as the shared parameters, whose dimension sizes are much smaller than that of the original update matrix $\boldsymbol{M}$. Since the most important information needed for model updating is carried by the top r ranks, the compression will have a marginal impact on the intensity of the useful signal.

However, decomposition of the model updates independently via full SVD is computation-intensive and not practical for resource-limited agents such as mobile devices. In addition, when agents calculate their own low-rank approximation, the server needs to obtain the values of individual $\boldsymbol{U}$ and $\boldsymbol{V}$ to reconstruct the approximate value of $\boldsymbol{M}$, which is not compatible with existing secure aggregation method in FL that only reveals the sum [10].

Based on the observation that only the average of the model updates needs to be obtained at the server, we propose to factorize the aggregated model updates instead of each individual model update via power iteration. Specifically, we only use one step of power iteration, i.e., the agent only performs a single step of subspace iteration to compute an approximation of the aggregated model update. Each step of subspace iteration involves two matrix multiplications and one orthogonalization. Since we divide the model updates into layers and factorize each layer independently, the matrix multiplication does not involve large matrices and can be computed efficiently. In the experiments, we use Gram-Schmidt procedure for the orthogonalization [9]. To improve the efficiency of power iteration, we use the approximation from the previous round as the starting point of the current round. Note that although agent sampling in FL appears to prevent the use of warm start due to the changing set of agents, it is feasible as the server already keeps a copy of the aggregated $\boldsymbol{U}$ and $\boldsymbol{V}$.

In the **SubspaceIterate** procedure, after obtaining the local model update $\boldsymbol{\Delta}_t^i$, the agent performs one-step subspace iteration with a right multiplication (line 26), bounds the norm of the result via clipping (line 27), and then perturbs the resulting subspace $\bar{\boldsymbol{U}}_{t+1}^i$ with Gaussian noise $\boldsymbol{b}_1 \sim \mathcal{MN}_{m\times r}(0, \mathbb{I}, \sigma_1^2\mathbb{I})$ (line 28). The perturbed results would be aggregated by the server and shared with agents to execute the **AgentUpdate** procedure (lines 8-13). In the **AgentUpdate** procedure, the agent performs one-step subspace iteration with a left multiplication (line 30), clips the result (line 31), and then perturbs the resulting matrice $\bar{\boldsymbol{V}}_{t+1}^i$ with Gaussian noise $\boldsymbol{b}_2 \sim \mathcal{MN}_{n\times r}(0, \mathbb{I}, \sigma_2^2\mathbb{I})$ (line 32). Next, all the perturbed matrices $\{\tilde{\boldsymbol{V}}_{t+1}^i, i \in \mathcal{S}_t\}$ will be aggregated by the server and used to reconstruct the model update $\boldsymbol{\Delta}_t$ for global model update (lines 14-17).

Note that power iteration does not change the sensitivities of $\boldsymbol{U}$ and $\boldsymbol{V}$ as they are determined by the corresponding clipping thresholds, which are fixed beforehand. Moreover, with the use of power iteration, the calculation of the aggregated $\boldsymbol{U}$ and $\boldsymbol{V}$ is linear, and secure aggregation can now be easily stacked with our approach. Secure aggregation hides the individual parameters and only reveals the aggregated $\boldsymbol{U}$ and $\boldsymbol{V}$, which enables us to reduce the magnitude of added Gaussian noise at each agent by a ratio of $\sqrt{S}$, where $S = |\mathcal{S}_t|$ is the number of selected agents at round t (see the proof of Theorem 1). In our algorithm, each agent encrypts its local $\boldsymbol{U}$ or $\boldsymbol{V}$ before sending it out, then the server sums the encrypted local subspaces (i.e., $\sum_{i\in\mathcal{S}_t} \mathcal{U}_{t+1}^i$) or embeddings (i.e., $\sum_{i\in\mathcal{S}_t} \mathcal{V}_{t+1}^i$) and decrypts it to obtain the sum of local subspaces (i.e., $\sum_{i\in\mathcal{S}_t} \tilde{\boldsymbol{U}}_{t+1}^i$) or embeddings (i.e., $\sum_{i\in\mathcal{S}_t} \tilde{\boldsymbol{V}}_{t+1}^i$). Note that the decryption happens after the ciphertexts of all agents at the current round have been aggregated, and thus individual plaintext is unknown to the server. We refer the readers to [10], [11] for the detailed encryption and decryption process of secure aggregation.

C. Privacy Analysis

The level of DP guarantee of the resulting learning algorithm depends on the values of noise magnitudes σ_1 and σ_2, as well as the sensitivities of perturbed matrices. To control the sensitivity, we follow the common practice to clip them with pre-defined thresholds C_1 and C_2. We can then use RDP to account the total privacy loss across T rounds. The final DP guarantee can be obtained by converting RDP back to DP, as given in Theorem 1.

Theorem 1 (Privacy Guarantee of CMP-Fed)**.** *Assume the noise magnitudes $\sigma_1 = C_1\sigma/\sqrt{S}$ and $\sigma_2 = C_2\sigma/\sqrt{S}$ where $\sigma > 0$ represents the noise multiplier, and the agent is sampled without replacement with probability $q := S/N$ at each round. After T rounds of training, CMP-Fed satisfies (ϵ, δ)-DP for any $\delta \in (0, 1)$, where*

$$\sigma^2 \geq \frac{14q^2T(\epsilon + 2\log(1/\delta))}{\epsilon^2},$$

if $\alpha \leq (2/3)\min\{C_1^2, C_2^2\}\sigma^2 \log(1/q\alpha(1+\sigma^2)) + 1$ and $\sigma^2 \geq 0.7$.

Proof: Since the server only knows the sum of local subspaces/embeddings, i.e., $\sum_{i\in\mathcal{S}_t} \bar{\boldsymbol{U}}_{t+1}^i$ and $\sum_{i\in\mathcal{S}_t} \bar{\boldsymbol{V}}_{t+1}^i$, due to the use of secure aggregation, we need to compute the privacy loss incurred from releasing the above aggregated matrices at each round. Assume the agent sets $\mathcal{S}_t$ and $\mathcal{S}'_t$ differ in one agent index c such that $\mathcal{S}'_t := \mathcal{S}_t \bigcup\{c\}$. For any two adjacent datasets $D := \{D_i\}_{i\in\mathcal{S}_t}$ and $D' := \{D_j\}_{j\in\mathcal{S}'_t} = \{D_i\}_{i\in\mathcal{S}_t} \bigcup D_c$, according to Definition 3, the ℓ_2-sensitivities of the sum of local matrices are

$$\psi_{\boldsymbol{U}} := \sup_{D,D'} \left\| \sum_{i\in\mathcal{S}_t} \bar{\boldsymbol{U}}_{t+1}^i(D_i) - \sum_{j\in\mathcal{S}'_t} \bar{\boldsymbol{U}}_{t+1}^j(D_j) \right\|_2,$$

$$\psi_{\boldsymbol{V}} := \sup_{D,D'} \left\| \sum_{i\in\mathcal{S}_t} \bar{\boldsymbol{V}}_{t+1}^i(D_i) - \sum_{j\in\mathcal{S}'_t} \bar{\boldsymbol{V}}_{t+1}^j(D_j) \right\|_2.$$

Due to the clipping, we have $\|\bar{\boldsymbol{U}}_{t+1}^i(D_i)\|_2 \leq C_1, \forall i \in [N]$ and $\|\bar{\boldsymbol{V}}_{t+1}^i(D_i)\|_2 \leq C_2, \forall i \in [N]$, and therefore $\psi_{\boldsymbol{U}} = \sup_{D,D'} \left\|\bar{\boldsymbol{U}}_{t+1}^c(D_c)\right\|_2 \leq C_1$ and $\psi_{\boldsymbol{V}} = \sup_{D,D'} \left\|\bar{\boldsymbol{V}}_{t+1}^c(D_c)\right\|_2 \leq C_2$. As the sum of Gaussian random variables is still a Gaussian random variable, the variance of the Gaussian noise added to each selected coordinate of the sum of local matrices are $S\sigma_1^2$ and $S\sigma_2^2$, respectively.

Let σ represent the noise multiplier, and assume that $\sigma_1 = C_1\sigma/\sqrt{S}$ and $\sigma_2 = C_2\sigma/\sqrt{S}$, then the aggregated matrices $\boldsymbol{U}_{t+1}$ and $\boldsymbol{V}_{t+1}$ at t-th round satisfy $(\alpha, \alpha/2\sigma^2)$-RDP and $(\alpha, \alpha/2\sigma^2)$-RDP, respectively (by Lemma 2). Then, the t-th round of Algorithm 1 satisfies $(\alpha, \alpha/\sigma^2)$-RDP according to Lemma 4. Suppose the agent is sampled without replacement with probability $q := S/N$ at each round. By Lemma 3, the t-th round of Fed-SMP satisfies $(\alpha, 7q^2\alpha/\sigma^2)$-RDP, if $\sigma^2 \geq 0.7$ and $\alpha \leq 1 + (2/3)\min\{C_1^2, C_2^2\}\sigma^2 \log(1/q\alpha(1+\sigma^2))$. Next, by Lemma 4, Fed-SMP satisfies $(\alpha, 7q^2T\alpha/\sigma^2)$-RDP after T rounds of training. Finally, in order to guarantee (ϵ, δ)-DP according to Lemma 1, we need

$$\frac{7q^2T\alpha}{\sigma^2} + \frac{\log(1/\delta)}{\alpha - 1} \leq \epsilon.$$

Choose $\alpha = 1 + 2\log(1/\delta)/\epsilon$ and rearrange the inequality in (III-C), we need

$$\sigma^2 \geq \frac{14q^2T(\epsilon + 2\log(1/\delta))}{\epsilon^2}.$$

■

Note that the calculation of privacy loss depends on the choice of the hyperparameter α, and in the experiments, we numerically compute the total privacy loss using the Opacus package in python [28].

IV. Experimental Evaluation

In this section, we evaluate the performance of CMP-Fed under different levels of compression, and compare it with the following baselines:

- FedAvg: the classic non-private federated learning algorithm;
- DP-FedAvg [22]: a classic differentially-private variant of FedAvg, where the local model update is first perturbed directly by adding Gaussian noise drawn from the distribution $\mathcal{N}(0, \sigma^2\mathbb{I}_d)$ (here d denotes the dimension of the original model update);
- DP-RandAgg: a communication efficient variant of DP-FedAvg, where the local model update is perturbed by adding Gaussian noise drawn from the distribution $\mathcal{N}(0, \sigma^2\mathbb{I}_d)$ and then sparsified using the random-k sparsification [29]. Note that for fair comparison, we adapt the proposed algorithm in [29] to be compatible with secure aggregation, where a common subset of coordinates is sampled at the beginning of each round and used by all selected agents in that round;
- UV-Fed: the non-private counterpart of CMP-Fed where the model updates are compressed via low-rank approximation without adding any DP noise.

All the algorithms are implemented in PyTorch, and all the experiments are executed on a workstation of 4 Nvidia Quadro RTX 8000 GPUs. When using one GPU, each experiment takes around 20 minutes.

A. Experimental Settings

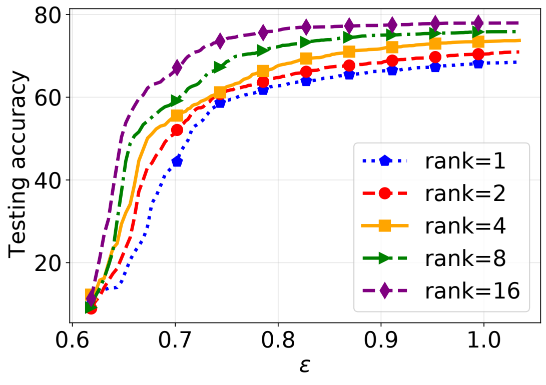
1) Datasets and Models: We conduct experiments on Fashion-MNIST [30], a common benchmark for differentially private machine learning. While the Fashion-MNIST is considered as “solved” in the computer vision community, achieving high utility with strong privacy guarantee remains difficult on this dataset [31]–[33]. The Fashion-MNIST dataset consists of 60,000 28×28 grayscale images of 10 fashion categories, along with a set of 10,000 test samples. To simulate the IID data distribution, we randomly split the training data among 6,000 agents. To simulate the non-IID data distribution, we split the training data among 6,000 agents such that images at each agent only cover 5 labels. We use the CNN model in [1], which consists of two 5×5 convolution layers (the first with 32 filters, the second with 64 filters, each followed with 2×2 max pooling), a fully connnected layer with 512 units and ReLu activation, and a final softmax output layer (1.6 million parameters in total).

2) Hyperparameters: For all experiments, we set the number of selected agents per round $S = 100$, number of communication rounds $T = 180$, local iteration period $K = 10$, and global learning rate $\gamma = 1.0$ by default. We use momentum SGD as the local optimizer of agents with a momentum coefficient of 0.5 and batch size of 10. The local learning rate is tuned over the grid $\eta = \{0.001, 0.01, 0.05, 0.125\}$ and decays at a rate of 0.99 at each round. It is worth mentioning that the momentum at each agent is initialized at every communication round since local momentum will be stale due to the partial participation of agents. For all privacy-preserving experiments, we set the privacy failure probability $\delta = 10^{-4}$ so that $\delta < 1/N$, and the noise magnitude σ for each private scheme is computed beforehand given the privacy budget ϵ. The clipping parameter is selected according to the median of the l_2-norm of the subspace/embedding/model to be clipped. Specifically, we have $C_1 = 0.01, C_2 = 1.0$ for CMP-Fed and $C = 1.0$ for DP-FedAvg and DP-RandAgg.

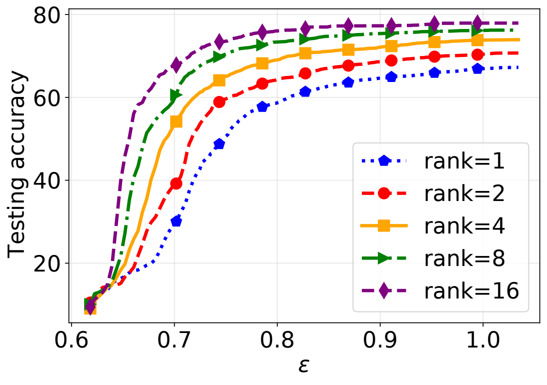
B. Experimental Results

1) Impact of compression level in CMP-Fed: In Fig. 1, we show the privacy-accuracy trade-off of CMP-Fed when using different coefficients r in the low-rank approximation corresponding to different levels of compression. It is easy to observe that as the privacy budget ϵ increases, more communication rounds are allowed and the testing accuracy improves accordingly. Besides, a small compression level results in less amount of added Gaussian noise and smaller privacy error, but larger compression error. From the figure, we can also observe that as the rank goes down, the compression error starts to dominate and affects the testing accuracy more than the privacy error brought by the DP noise. Thus the accuracy was reduced as the rank reduces from 16 to 1. This has been observed under both IID and non-IID settings.

2) Testing accuracy comparison: We compare the testing accuracy of CMP-Fed with non-private baselines FedAvg [1],



(a) IID setting.



(b) Non-IID setting.

Fig. 1: Privacy-accuracy trade-off of CMP-Fed under different compression levels

Algorithm	Testing accuracy (%)					
	rank = 1	rank = 2	rank = 4	rank = 8	rank = 16	no compression
CMP-Fed	68.54 ± 2.41	71.00 ± 1.09	73.64 ± 0.29	76.20 ± 0.90	78.87 ± 0.05	-
DP-RandAgg	52.42 ± 2.76	59.05 ± 3.91	66.82 ± 2.39	70.93 ± 0.89	74.84 ± 0.81	-
UV-Fed	77.44 ± 0.67	79.71 ± 0.64	83.10 ± 0.23	84.76 ± 0.18	86.15 ± 0.26	-
DP-FedAvg	-	-	-	-	-	77.65 ± 0.46
FedAvg	-	-	-	-	-	86.61 ± 0.40

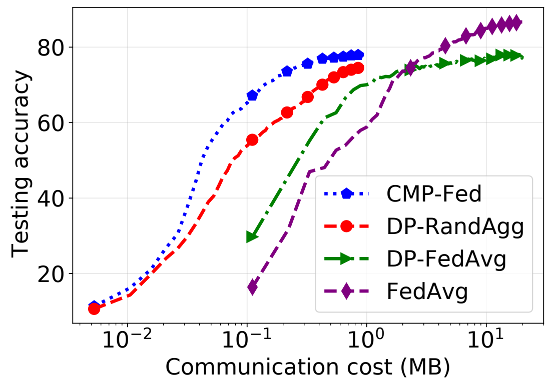
TABLE I: Summary of results on Fashion-MNIST dataset (IID). CMP-Fed, DP-RandAgg, and DP-FedAvg achieve $(1, 10^{-4})$-DP, and UV-Fed and FedAvg are non-private baselines.

Algorithm	Testing accuracy (%)					
	rank = 1	rank = 2	rank = 4	rank = 8	rank = 16	no compression
CMP-Fed	67.25 ± 2.93	70.72 ± 1.87	73.92 ± 1.03	76.24 ± 0.74	77.93 ± 0.62	-
DP-RandAgg	49.67 ± 4.76	56.56 ± 2.98	65.54 ± 2.01	69.89 ± 1.67	74.12 ± 0.64	-
UV-Fed	77.15 ± 0.31	80.24 ± 0.71	82.87 ± 0.41	84.68 ± 0.19	85.82 ± 0.26	-
DP-FedAvg	-	-	-	-	-	75.75 ± 0.86
FedAvg	-	-	-	-	-	86.49 ± 0.44

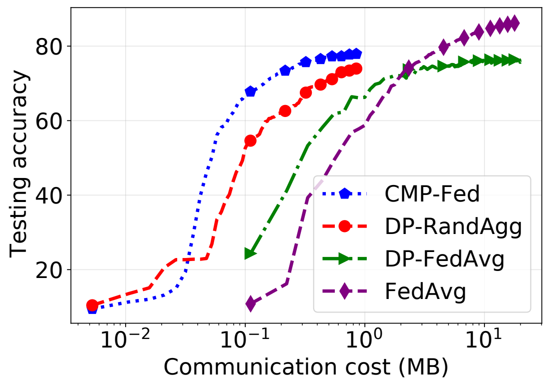
TABLE II: Summary of results on Fashion-MNIST dataset (non-IID). CMP-Fed, DP-RandAgg, and DP-FedAvg achieves $(1, 10^{-4})$-DP, and UV-Fed and FedAvg are non-private baselines.

DP-FedAvg [22], and a modified version of prior work, DP-RandAgg [29]. The results are summarized in Table I (IID setting) and Table II (non-IID setting). Note that we always let DP-RandAgg achieve the same compression ratio of CMP-Fed which is $r(m+n)/mn$. We also add the low-rank approximation algorithm UV-Fed for illustrative purpose, which is a non-private version of CMP-Fed. We run each experiment with 5 random seeds and report the average and standard deviation. We can observe that in both IID and non-IID settings, the non-private baseline FedAvg has a testing accuracy of 86% after 180 rounds. For algorithms with agent-level DP, Gaussian noises are added in the training process, which inevitably degrades the accuracy. Specifically, the accuracy of DP-FedAvg decreases to 78% under IID setting and 76% under non-IID setting with $(1.0, 10^{-4})$-DP.

Our basic intuition that CMP can improve privacy-accuracy tradeoffs has been verified in the table as well. Specifically, CMP-Fed can reach higher accuracy than DP-FedAvg while achieving the same privacy guarantee, e.g., when $r = 16$, CMP-Fed improves the accuracy by 1.6% in IID setting and 2.9% in non-IID setting respectively, compared with DP-FedAvg. This is due to the fact that compression reduces the noise magnitude introduced by DP and boosts the signal-to-noise ratio in the noisy model update. However, as r decreases, the compression error starts to dominate and offset the privacy amplification effect, which decreases the testing accuracy of CMP-Fed. This can be inferred from the comparison between UV-Fed and CMP-Fed when $r = 1$: CMP-Fed drops by 12% in accuracy from DP-FedAvg in IID setting, and the testing accuracy of UV-Fed also drops by 11% from FedAvg; and in non-IID setting, CMP-Fed drops by 11% in accuracy from DP-FedAvg and UV-Fed drops by 11% in accuracy from FedAvg.



(a) IID setting.



(b) Non-IID setting.

Fig. 2: Accuracy v.s. communication cost comparison for CMP-Fed and baselines.

3) Communication efficiency comparison: We also compare CMP-Fed with other baselines in terms of communication efficiency. Considering that uplink traffic is typically much more expensive than downlink traffic [1], we estimate the communication cost for the involved algorithms by counting the size of the data sent from a agent to the server as the input into the secure aggregation protocol. For CMP-Fed and UV-Fed, the uplink data size equals to $32r(m+n) \times T \times (S/N)$ bits; the uplink data size for DP-RandAgg is also $32r(m+n) \times T \times (S/N)$ bits; and the uplink cost for DP-FedAvg and FedAvg is $32mn \times T \times (S/N)$ bits. From this comparison, it is easy to see that CMP-Fed and DP-RandAgg have lower uplink communication cost compared to DP-FedAvg with the same communication rounds. Note that with secure aggregation protocols [10], [11], the actual traffic would be larger. However, this is not the focus of this paper and does not impact the conclusions we draw here.

We plot the testing accuracy of FedAvg, DP-FedAvg, CMP-Fed with $r = 16$, and DP-RandAgg with the same compression level as CMP-Fed, with respect to the uplink communication cost in Fig. 2. We can observe that under IID setting, to achieve a target accuracy 74%, CMP-Fed saves 69%, 90% and 89% uplink communication cost compared with DP-RandAgg, DP-FedAvg and FedAvg, respectively; under non-IID setting, CMP-Fed saves 73%, 93% and 89% uplink communication cost compared with DP-RandAgg, DP-FedAvg and FedAvg, respectively.

V. Related Works

DP has been widely used for providing rigorous privacy guarantee in machine learning [12], [34]–[39]. General DP mechanisms, such as Gaussian or Laplacian mechanism, rely on the addition of carefully calibrated noise to the output of an algorithm directly. The additive noise is proportional to the model size, which would be prohibitively large (e.g., millions of model parameter) for DNNs commonly used in FL. To boost the utility-privacy trade-off, two prior work have proposed to use gradient compression for improving privacy-utility trade-off in centralized learning setting [40], [41]. However, both work require a non-private public dataset to identify the subspace for gradient compression [40], [41], which is difficult to obtain for FL settings with non-IID data distribution. In addition, [40] requires the transmission of full-dimension residue, which is not communication-efficient. The work closest to ours is [29], which uses random sparsification for privacy amplification in the FL setting. However, it only considers record-level DP, and the proposed scheme will lose the communication benefit of sparsification if directly combined with secure aggregation for agent-level DP. We have compared our approach with a modified version of [29] that is compatible with secure aggregation, and shown that our approach can achieve a better privacy-accuracy trade-off.

It is worth noting that besides improving utility-privacy trade-offs, our scheme is also communication-efficient by compressing the shared model updates in FL. While there have been a line of work on improving communication efficiency in distributed learning [6]–[9], [42], none of them jointly consider privacy and communication efficiency. The closest paper to ours is cpSGD [15], which is a modified distributed SGD scheme that provides agent-level DP and communication efficiency simultaneously via combining gradient quantization and Binomial mechanism. However, their model accuracy has been reported to be much worse than other baselines such as DP-RandAgg in [29] because quantization may increase rather than decrease the intensity of additive DP noise, and therefore we do not include their results in this paper.

VI. Conclusion

In this paper, we have proposed CMP-Fed, a novel differentially private FL scheme via compressed model perturbation to achieve agent-level DP with high model accuracy in FL. Our choices of linear compressor, speed-up choice, as well as other key designs enable us to greatly improve the accuracy and applicability of CMP-Fed. Experimental results have demonstrated the superior performance of our approach compared with prior approaches.

Acknowledgment

The work of R. Hu and Y. Gong was supported in part by the U.S. National Science Foundation under grants CNS-2047761 and CNS-2106761. The work of Y. Guo was supported in part by the US NSF under grants CNS-2106761 and CMMI-2222670.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[2] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[3] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1322–1333.

[4] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy*, 2017, pp. 3–18.

[5] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[6] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, "Atomo: Communication-efficient learning via atomic sparsification," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[7] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," in *International Conference on Machine Learning*. PMLR, 2018, pp. 560–569.

[8] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[9] T. Vogels, S. P. Karimireddy, and M. Jaggi, "PowerSGD: Practical low-rank gradient compression for distributed optimization," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[10] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[11] J. H. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova, "Secure single-server aggregation with (poly) logarithmic overhead," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1253–1269.

[12] R. Hu, Y. Guo, and Y. Gong, "Concentrated differentially private federated learning with performance analysis," *IEEE Open Journal of the Computer Society*, vol. 2, pp. 276–289, 2021.

[13] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Advances in Neural Information Processing Systems*, 2018, pp. 1299–1309.

[14] T. Li, Z. Liu, V. Sekar, and V. Smith, "Privacy for free: Communication-efficient learning with differential privacy using sketches," *arXiv preprint arXiv:1911.00972*, 2019.

[15] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and B. McMahan, "cpSGD: Communication-efficient and differentially-private distributed SGD," in *Neural Information Processing Systems*, 2018.

[16] J. Ding, G. Liang, J. Bi, and M. Pan, "Differentially private and communication efficient collaborative learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[17] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang, "Stronger generalization bounds for deep nets via a compression approach," in *International Conference on Machine Learning*. PMLR, 2018, pp. 254–263.

[18] C. H. Martin and M. W. Mahoney, "Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning," *Journal of Machine Learning Research*, vol. 22, no. 165, pp. 1–73, 2021.

[19] I. Mironov, "Renyi differential privacy," in *IEEE Computer Security Foundations Symposium*, 2017, pp. 263–275.

[20] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1226–1235.

[21] L. Wang, B. Jayaraman, D. Evans, and Q. Gu, "Efficient privacy-preserving nonconvex optimization," *arXiv preprint arXiv:1910.13659*, 2019.

[22] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *International Conference on Learning Representations*, 2018.

[23] S. Goryczka, L. Xiong, and V. Sunderam, "Secure multiparty aggregation with differential privacy: A comparative study," in *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pp. 155–163.

[24] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 1–11.

[25] F. Valovich and F. Alda, "Computational differential privacy from lattice-based cryptography," in *International Conference on Number-Theoretic Methods in Cryptology*. Springer, 2017, pp. 121–141.

[26] K. Bonawitz, F. Salehi, J. Konečný, B. McMahan, and M. Gruteser, "Federated learning with autotuned communication-efficient secure aggregation," in *Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 1222–1226.

[27] W.-N. Chen, C. A. Choquette-Choo, and P. Kairouz, "Communication efficient federated learning with secure aggregation and differential privacy," in *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021.

[28] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Ghosh, A. Bharadwaj, J. Zhao, G. Cormode, and I. Mironov, "Opacus: User-friendly differential privacy library in PyTorch," *arXiv preprint arXiv:2109.12298*, 2021.

[29] R. Hu, Y. Gong, and Y. Guo, "Federated learning with sparsification-amplified privacy and adaptive optimization," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021, pp. 1463–1469.

[30] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms.

[31] N. Papernot, A. Thakurta, S. Song, S. Chien, and U. Erlingsson, "Tempered sigmoid activations for deep learning with differential privacy," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, 2021, pp. 9312–9321.

[32] F. Tramer and D. Boneh, "Differentially private learning needs better features (or much more data)," in *International Conference on Learning Representations*, 2021.

[33] D. Chen, T. Orekondy, and M. Fritz, "GS-WGAN: A gradient-sanitized approach for learning differentially private generators," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 673–12 684, 2020.

[34] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[35] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet of Things Journal*, 2020.

[36] Z. Huang, R. Hu, Y. Guo, E. Chan-Tin, and Y. Gong, "DP-ADMM: ADMM-based distributed learning with differential privacy," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1002–1012, 2019.

[37] Y. Guo and Y. Gong, "Practical collaborative learning for crowdsensing in the internet of things with differential privacy," in *IEEE Conference on Communications and Network Security*, 2018, pp. 1–9.

[38] Y. Gong, Y. Fang, and Y. Guo, "Private data analytics on biomedical sensing data via distributed computation," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 13, no. 3, pp. 431–444, 2016.

[39] R. Hu and Y. Gong, "Trading data for learning: Incentive mechanism for on-device federated learning," in *IEEE Global Communications Conference*, 2020, pp. 1–6.

[40] Y. Zhou, S. Wu, and A. Banerjee, "Bypassing the ambient dimension: Private SGD with gradient subspace identification," in *International Conference on Learning Representations*, 2021.

[41] D. Yu, H. Zhang, W. Chen, and T.-Y. Liu, "Do not let privacy overbill utility: Gradient embedding perturbation for private learning," in *International Conference on Learning Representations*, 2021.

[42] Y. Guo, Y. Sun, R. Hu, and Y. Gong, "Hybrid local SGD for federated learning with heterogeneous communications," in *International Conference on Learning Representations*, 2021.