Differentially Private Federated Learning with Local Regularization and Sparsification

Anda Cheng[1,2] Peisong Wang[1] Xi Sheryl Zhang[1] Jian Cheng[1,2]*
[1]Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
chenganda2017@ia.ac.cn sheryl.zhangxi@gmail.com {peisong.wang,jcheng}@nlpr.ia.ac.cn

Abstract

User-level differential privacy (DP) provides certifiable privacy guarantees to the information that is specific to any user's data in federated learning. Existing methods that ensure user-level DP come at the cost of severe accuracy decrease. In this paper, we study the cause of model performance degradation in federated learning with user-level DP guarantee. We find the key to solving this issue is to naturally restrict the norm of local updates before executing operations that guarantee DP. To this end, we propose two techniques, Bounded Local Update Regularization and Local Update Sparsification, to increase model quality without sacrificing privacy. We provide theoretical analysis on the convergence of our framework and give rigorous privacy guarantees. Extensive experiments show that our framework significantly improves the privacy-utility trade-off over the state-of-the-arts for federated learning with user-level DP guarantee.

1. Introduction

Federated learning (FL) [17] is a promising paradigm of distributed machine learning with a wide range of applications [5, 13, 15]. FL enables distributed agents to collaboratively learn a centralized model under the orchestration of the cloud without sharing their local data. By keeping data usage local, FL sidesteps the ethical and legal concerns and is advantageous in privacy compared with the traditional centralized learning paradigm.

However, FL alone does not protect the agents or users from inference attacks that use the output information. Extensive inference attacks demonstrate that it is feasible to infer the subgroup of people with a specific property [19], identify individuals [24], or even infer completion of social security numbers [4], with high confidence from a trained model.

*Corresponding Author.

To solve these issues, differential privacy (DP) [6] has been applied to FL in order to protect either each instance in the dataset of any agent (instance-level DP) [11, 25, 26], or the whole data of any agent (user-level DP) [7, 12, 18]. These two DP definitions on different levels are suitable for different situations. For example, when several banks aim to train a fraud detection model via FL, instance-level DP is more suitable to protect any individual records of any bank from being identified. In another situation, when a smartphone app attempts to learn a face recognition model from users' face images, it is more appropriate to apply user-level DP to protect each user as a unit.

Existing methods that ensure user-level DP [7, 12, 18] are predominantly built upon Gaussian mechanism which is a Gaussian noise perturbation-based technique. Unfortunately, directly applying the Gaussian mechanism to ensure strong user-level DP in FL drastically degrades the utility of the resulted models. Specifically, the Gaussian mechanism requires to clip the l_2 magnitude of local updates to a sensitivity threshold S and adding noise proportional to S to the high dimensional local updates. These two steps lead to either large bias (when S is small) or large variance (when S is large), which slows down the convergence and damages the performance of the global model [30]. However, existing methods [7, 12, 18] do not explicitly involve interaction between the operations for ensuring DP and the learning process of FL, which makes the learning process hard to adapt to the clipping and noise perturbation operations, thereby leading to utility degradation of the learned models.

To address the above issues, in this paper, we propose two techniques to improve the model utility in FL with user-level DP guarantees. Our motivation is to naturally reduce the l_2 norm of local updates before clipping, thereby making the local updates more adaptive to the clipping operation. First, we propose *Bounded Local Update Regularization (BLUR)*. It introduces a regularization term to the agent's local objective function and explicitly regularizes the l_2 norm of local updates to be bounded. As a result, the

l_2 norm of local updates could be naturally smaller than S, thereby decreasing the impact of clipping operation. Then we propose *Local Update Sparsification (LUS)* to further reduce the magnitude of local updates. Before clipping, it zeros out some update values that have little effect on the performance of the local model, thereby reducing the norm of local updates without damaging the accuracy of the local model.

Our contributions can be summarized as follows:

- We propose two techniques to improve the model utility with user-level DP guarantee in federated learning.
- We provide theoretical analysis on the convergence of our framework and give rigorous privacy guarantees.
- Extensive experiments validate the effectiveness and advantages of the proposed methods.

2. Related Work

The concept of user-level differential privacy in federated learning was introduced by [18]. They propose DP-FedAvg to train models for next-word prediction in a mobile keyboard meanwhile ensuring user-level DP guarantee by employing Gaussian mechanism and composing privacy guarantees via moment accountant. The following work [12, 27] ensures user-level DP by discretizing the data and adding discrete Gaussian noise before performing secure aggregation. They also provide a novel privacy analysis for sums of discrete Gaussians. Both of the above methods ensure user-level DP via noise perturbation-based method, which requires to clip norm of model update or data and add noise to the clipped vectors. Nevertheless, the clipping and noise perturbation steps inevitably interfere with the performance of the resulting model. Different from the aforementioned methods, a recent study [30] proposed AE-DPFL which ensured user-level DP by a voting-based mechanism with secure aggregation. AE-DPLF does not need to clip the model or data, thereby relieving the accuracy degradation issue. However, the AE-DPFL framework assumes that unlabeled data from the global distribution is available to the server, which is very hard to satisfy in practical applications. Our work follows the paradigm of the noise perturbation methods but we aim to improve the training process by naturally bounding the local update norms.

Other works related to our paper are those employing regularization or sparsification techniques in FL. Previous works [22] and [2] also introduce regularization terms into objective function for each device. Nevertheless, they aim to apply the regularization technique to address the data/device distribution heterogeneity issue in FL, which is different from our goal of bounding the sensitivity of local updates. Another line of works [3, 11] also apply the sparsification technique in privacy-preserving FL. Both of them focus on ensuring instance-level DP and employing sparsification to reduce communication costs. On the contrary, our work utilizes the sparsification technique to improve the model utility with user-level DP guarantee.

3. Preliminary

3.1. Federated Learning (FL)

Federated learning [17] is a multi-round protocol between an aggregation server and a set of agents in which agents jointly train a model. Let $\mathcal{P}$ denotes the set of all agents with $|\mathcal{P}| = N$, and $\mathcal{D}_i$ denote the local dataset of client $i \in \mathcal{P}$ with n_i samples. The set $\mathcal{D} = \bigcup_{i\in\mathcal{P}} \mathcal{D}_i$ denotes the full training set. Let $f_i(\mathbf{w}, z)$ denotes the loss function for client i over a model $\mathbf{w}$ and a sample z, and $f_i(\mathbf{w}, \mathcal{D}_i) = \frac{1}{n_i}\sum_{z\in\mathcal{D}_i} f_i(\mathbf{w}, z)$ denotes the empirical loss over a model $\mathbf{w}$ and a dataset $\mathcal{D}_i$. Without causing ambiguity, we also denote the local loss function as $f_i(\mathbf{w})$ in the following. In FL, agents try to jointly train a model that minimizes the weighted average of local loss functions:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \{f(\mathbf{w}, \mathcal{D}) = \sum_{i\in\mathcal{P}} \frac{n_i}{n} f_i(\mathbf{w}, \mathcal{D}_i)\} \tag{1}$$

where $n = \sum_{i\in\mathcal{P}} n_i$ is the total dataset size of all agents. To solve this optimization task, the widely used FedAvg protocol executes the following two steps in the communication round t:

- Local updating. The server samples a set of agents $\mathcal{P}_t$. Each agent $i \in \mathcal{P}_t$ downloads the global model $\mathbf{w}^{t-1}$ from server, then performs local training on local dataset by executing $\mathbf{w}_i^{t,q} \leftarrow \mathbf{w}_i^{t,q-1} - \eta_l \nabla_{\mathbf{w}} f_i\left(\mathbf{w}_i^{t,q-1}, \mathcal{D}_i\right)$ for Q steps with $\mathbf{w}_i^{t,0}$ initialized as $\mathbf{w}^{t-1}$. Finally, each agent uploads the model update $\Delta_i^t = \mathbf{w}_i^{t,Q} - \mathbf{w}_i^{t,0}$ to server.
- Model aggregation. The server receives model updates $\{\Delta_i^t | i \in \mathcal{P}_t\}$ from participants and aggregates them to update global model by $\mathbf{w}^t \leftarrow \mathbf{w}^{t-1} + \eta_g \sum_{i\in\mathcal{P}_t} \frac{n_i}{n_{\mathcal{P}_t}} \Delta_i^t$.

3.2. Differential Privacy (DP)

Differential privacy [6] is a formal notion of privacy that provides provable guarantees against the identification of individuals in a private set. We denote $D \simeq D'$ as a pair of *adjacent datasets*, which means that D' can be obtained from D by changing only one record.

Definition 1 ***Differential Privacy.*** *A randomized algorithm $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy if for any adjacent datasets $D \simeq D'$ for any subset of outputs $\mathcal{S} \subseteq Range(\mathcal{M})$ it holds that $\Pr[\mathcal{M}(D) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in S] + \delta$.*

The definition indicates that one could not distinguish between D and D' by observing the output of $\mathcal{M}$, thereby protecting individuals in D from identification. A simple way to achieve (ϵ, δ)-DP is to take a vector-valued deterministic function F and inject appropriate Gaussian noise, the scale of which depends on the sensitivity of F.

Definition 2 (l_2 Sensitivity). *Let $\mathcal{F}$ be a function, the L_2-sensitivity of $\mathcal{F}$ is defined as $\mathcal{S} = \max_{D \simeq D'} \|\mathcal{F}(D) - \mathcal{F}(D')\|_2$, where the maximization is taken over all pairs of adjacent datasets.*

Lemma 1 *Let $\mathcal{F}$ be a function, $\delta \in (0,1)$ and $\epsilon > 0$. For $c > \sqrt{2\ln(1.25/\delta)}$, the Gaussian mechanism $\mathcal{F}(\mathcal{D}) + \mathcal{N}(0, \sigma^2\mathbf{I})$ with parameter $\sigma \geq c\mathcal{S}/\epsilon$ ensures (ϵ, δ)-DP.*

3.3. Differential Privacy for Federated Learning

In FL, DP can be defined into *instance-level DP* and *user-level DP*, depending on how adjacent dataset is defined. Our work focuses on the latter.

Definition 3 ***User-level DP***. *When $\mathcal{D}'$ is constructed by adding or removing **one agent** with all its data records.*

DP-FedAvg [18] is the first to guarantee user-level DP in FL by applying Gaussian mechanism. To ensure user-level DP, before uploading local updates to the server, DP-FedAvg clips the norm of per-agent model update Δ_i^t to a threshold S and adds scaled Gaussian noise to the bounded update, as shown in Alg. 1. Although DP-FedAvg ensures user-level DP, it severely harms the utility of the resulted models. In this work, we aim to develop a federated learning framework that has little negative impact on the model utility meanwhile ensuring user-level DP.

Algorithm 1 DP-FedAvg

Input: Agent sampling probability $p \in (0, 1]$, clipping threshold S, noise scale σ.
Output: Trained model $\mathbf{w}^T$
Server

Initialize global model $\mathbf{w}_0$
for $t = 1$ to T **do**
  $\mathcal{P}_t \leftarrow$ Sample agents with probability p;
  for $i \in \mathcal{P}_t$ in parallel **do**
    $\overline{\Delta}_i^t = \text{LocalUpdate}\left(\mathbf{w}^{t-1}, i\right)$;
  end for
  $\mathbf{w}^t \leftarrow \mathbf{w}^{t-1} + \eta_g \frac{1}{|\mathcal{P}_t|} \sum_{i \in \mathcal{P}_t} \overline{\Delta}_i^t$;
end for
return $\mathbf{w}^T$

LocalUpdate

$\mathbf{w}_i^{t,0} \leftarrow$ Download $\mathbf{w}^{t-1}$;
for $q = 1$ to Q **do**
  Sample batch $\mathcal{B} \subseteq \mathcal{D}_i$;
  $\mathbf{w}_i^{t,q} \leftarrow \mathbf{w}_i^{t,q-1} - \eta_l \frac{1}{|\mathcal{B}|} \sum_{(x,y) \in \mathcal{B}} \nabla_{\mathbf{w}} f_i\left(\mathbf{w}_i^{t,q-1}, x, y\right)$;
end for
$\Delta_i^t = \mathbf{w}_i^{t,Q} - \mathbf{w}_i^{t,0}$;
$\widetilde{\Delta}_i^t = \Delta_i^t / \max\left(1, \frac{\|\Delta_i^t\|_2}{S}\right)$;
return $\widetilde{\Delta}_i^t + \mathcal{N}(0, S^2\sigma^2\mathbf{I}_d / |\mathcal{P}_t|)$

4. Methodology

We start by analyzing the impact of clipping and adding noise operations in the local update. We denote Δ_i^t as the local update at communication round t from agent i before clipping, denote $\widetilde{\Delta}_i^t$ as the local update after clipping but before adding noise, and denote $\overline{\Delta}_i^t$ as the local update after clipping and adding noise. Let d denote the dimension of Δ_i^t, then the expected *mean-square error* of the estimate $\overline{\Delta}_i^t$ can be computed as follows

$$\begin{aligned}\mathbb{E}\left[\frac{1}{d}\left\|\overline{\Delta}_i^t - \Delta_i^t\right\|_2^2\right] &\leq \frac{1}{d}\left(\mathbb{E}\left[\left\|\widetilde{\Delta}_i^t - \Delta_i^t\right\|_2^2 + \left\|\overline{\Delta}_i^t - \widetilde{\Delta}_i^t\right\|_2^2\right]\right) \\ &= \frac{1}{d}\max\left(0, \|\Delta_i^t\| - S\right)^2 + \frac{\sigma^2 S^2}{|\mathcal{P}_t|}\end{aligned} \tag{2}$$

The detailed derivation of Eq. 2 is provided in the Appendix. Eq. 2 indicates that $\overline{\Delta}_i^t$ is a biased estimation of Δ_i^t. At the right side of Eq. 2, the first and second term reflect the deviation introduced by clipping and adding Gaussian noise, respectively. To minimize the deviation, we can decrease the right side of the inequality in two ways:

- Ensuring $\|\Delta_i^t\|$ is not greater than S for each i and t;
- Using a smaller clipping threshold S.

The first way indicates that we should somehow limit the l_2 norm of the local update to make it smaller than S. Intuitively, if $\|\Delta_i^t\|$ is large, e.g. $\|\Delta_i^t\| \gg S$, the clipping operation could lead to much of the update information contained in Δ_i^t be dropped and makes the resulted $\widetilde{\Delta}_i^t$ less informative. The second way indicates that we can use smaller S to limit the impact of Gaussian noise. Intuitively, this works because the variance of added Gaussian noise is proportional to S^2. Using smaller S can directly reduce the perturbation effect of adding noise. However, when we also consider the first way, we can find that it is difficult to reduce the deviation by only reducing S without considering $\|\Delta_i^t\|$. Because for the same $\|\Delta_i^t\|$ that is greater than S, only reducing S could increase $\|\Delta_i^t\| - S$, which enlarges the negative impact of clipping operation. This indicates that the key to solve the problem is to **naturally reduce the norm of local updates** at each communication round.

Based on the above observation, we propose two techniques to improve the utility federated learning with user-

level DP guarantee, termed *Bounded Local Update Regularization* and *Local Update Sparsification*. Our motivation is to reduce the norm of local updates by regularizing local models and making the local updates sparse.

4.1. Bounded Local Update Regularization (BLUR)

In vanilla FedAvg, each agent trains the local model by optimizing the objective function of

$$\min_{\mathbf{w}\in\mathbb{R}^d} f_i(\mathbf{w}) \tag{3}$$

which does not impose any constraints on weight updates. However, when we apply the Gaussian mechanism to ensure user-level DP, the l_2 norm of weight update must be limited to ensure the sensitivity of weight update smaller than a threshold S. To this end, the l_2 norm of weight update should be considered as a constraint in the local optimization. Let $\mathbf{w}^t$ denote the local initial weight at communication round t. Then the local optimization should be formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^d} f_i\left(\mathbf{w}\right) \quad \text{s.t. } \|\mathbf{w}-\mathbf{w}^t\| \leq S \tag{4}$$

The above formulation can be converted to an unconstrained optimization by transforming the constraint to a regularization term (BLUR) as

$$\begin{aligned} &\min_{\mathbf{w}\in\mathbb{R}^d}\{h_i(\mathbf{w}) \triangleq f_i(\mathbf{w}) + \frac{\lambda}{2}R_t(\mathbf{w})\} \\ &\text{where } R_t(\mathbf{w}) = \max\left(0, \|\mathbf{w}-\mathbf{w}^t\|^2 - S^2\right) \end{aligned} \tag{5}$$

Directly optimizing Eq. 3 may lead to $\|\Delta_i^t\| \gg S$, in which case applying clipping operation to Δ could result in much of the information in Δ_i^t being dropped, thereby impeding the convergence of the local training process. On the contrary, Δ_i^t obtained by optimizing Eq. 5 is more adaptive to the clipping operation as the regularization term in Eq. 5 effectively limits the l_2 sensitivity of Δ to be smaller than the clipping threshold S.

The effect of BLUR can also be interpreted as an adaptive adjustment to the local learning rate by considering both model update norm and learning step. Without using BLUR, the local update can be expressed as

$$\mathbf{w}_i^{t,Q} - \mathbf{w}^t = -\eta_l \sum_{q=0}^{Q-1} \mathbf{g}_i^{t,q} \tag{6}$$

where $\mathcal{B}_q$ denotes the local batch of data at the local step q and $\mathbf{g}_i^{t,q} = \frac{1}{|\mathcal{B}_q|}\sum_{(x,y)\in\mathcal{B}_q} \nabla f_i\left(\mathbf{w}_i^{t,q-1}, x, y\right)$ with $\mathbb{E}\left[\mathbf{g}_i^{t,q}\right] = \nabla f_i\left(\mathbf{w}_i^{t,q}\right)$. The result in Eq. 6 can be easily obtained by unrolling the update step of DP-FedAvg (line of LocalUpdate in Alg. 1). While applying BLUR, the local model is updated by optimizing Eq. 5. as

$$\mathbf{w}_i^{t,q} \leftarrow \mathbf{w}_i^{t,q-1} - \eta_l \frac{1}{|\mathcal{B}|}\sum_{(x,y)\in\mathcal{B}} \nabla h_i\left(\mathbf{w}_i^{t,q-1}, x, y\right) \tag{7}$$

Lemma 2 *Suppose at communication round t, the local model on agent i is updated by repeating Eq. 7 with $\lambda < \frac{1}{\eta_l}$ for Q iterations. Then we have the final local update*

$$\mathbf{w}_i^{t,Q} - \mathbf{w}^t = -\eta_l \sum_{q=0}^{Q-1} \gamma_i^{t,q} \mathbf{g}_i^{t,q} \tag{8}$$

$$\textit{where } \gamma_i^{t,q} = \begin{cases} (1-\lambda\eta_l)^q, & \textit{if } \|\mathbf{w}_i^{t,q} - \mathbf{w}^t\| > S \\ 1, & \textit{otherwise} \end{cases}$$

Lemma 2 shows that BLUR introduces an adaptive discount factor $\gamma_i^{t,q}$ to the local learning rate. At the local step q, if the norm of the current update is larger than S, the learning rate at this step would be discounted by $(1-\lambda\eta_l)^q$ to restrict the impact of this update step. On the contrary, if the norm of the current update is smaller than S, the effect of this step would not be limited. More concretely, the training process is forced to the local optimal that lies in the norm-restricted space.

We note that a similar regularization term has been applied in the previous work FedProx [22]. However, an important distinction between FedProx and our BLUR is that we aim to employ the regularization method to bound the sensitivity of the local updates by S, while FedProx applies the regularization method to tackle the statistical heterogeneity problem in federated learning. As a result, the impact of clipping threshold S is taken into account in our BLUR while FedProx does not involve a threshold in the regularization term.

4.2. Local Update Sparsification (LUS)

Sparsification is a widely used technique to improve communication efficiency in distributed training [16, 23, 28] or to reduce the model complexity of DNNs [8, 14]. Inspired by the previous works, we expect to further reduce the norm of local updates by eliminating some parameter updates which can be removed with less impact on model performance.

Suppose in a local update process, the initial model weight is $\mathbf{w_0}$. The model weight after local training is $\mathbf{w}$ and the corresponding update is $\Delta\mathbf{w}$. Here, we denote the whole model weight vector as $\mathbf{w}$ and denote a specific parameter in the model as w. We can zero out the update of a specific parameter w to 0 by setting $w \leftarrow w_0$ and get the corresponding model weight $\widetilde{\mathbf{w}}$ and model update $\Delta\widetilde{\mathbf{w}}$. By applying the Taylor series on $f_i(\widetilde{\mathbf{w}})$, we can get the loss value as

$$f_i\left(\widetilde{\mathbf{w}}\right) = f_i\left(\mathbf{w}\right) - \frac{\partial f_i\left(\mathbf{w}\right)}{\partial w}(w_0 - w) + o\left(w^2\right) \tag{9}$$

Ignoring the higher-order term, we have

$$\left|f_i\left(\widetilde{\mathbf{w}}\right) - f_i\left(\mathbf{w}\right)\right| = \left|\frac{\partial f_i\left(\mathbf{w}\right)}{\partial w}(w_0 - w)\right| \tag{10}$$

Algorithm 2 Local Update with BLUR and LUS

Input: Current global model $\mathbf{w}^{t-1}$, clipping threshold S, noise scale σ, regularization factor λ, number of preserved update values s

Output: Local update

$\mathbf{w}_i^{t,0} \leftarrow$ Download $\mathbf{w}^{t-1}$;
for $q = 1$ to Q **do**
  Sample batch $\mathcal{B} \subseteq \mathcal{D}_i$;
  Update local model $\mathbf{w}_i^{t,q}$ using Eq. 7;
end for
$\Delta_i^t = \mathbf{w}_i^{t,Q} - \mathbf{w}_i^{t,0}$;
Compute mask matrix $M(\Delta_i^t, s)$ according to Eq. 12;
$\hat{\Delta}_i^t \leftarrow M(\Delta_i^t, s) \circ \Delta_i^t$;
$\widetilde{\Delta}_i^t = \hat{\Delta}_i^t / \max\left(1, \frac{\|\hat{\Delta}_i^t\|_2}{S}\right)$;
return $\widetilde{\Delta}_i^t + \mathcal{N}(0, S^2\sigma^2/|\mathcal{P}_t|)$

We define the utility cost of zeroing out Δw as

$$T(\Delta w; \mathbf{w}) \triangleq \left| \frac{\partial f_i(\mathbf{w})}{\partial w} \Delta w \right| = \left| \frac{\partial f_i(\mathbf{w})}{\partial w} (w_0 - w) \right| \quad (11)$$

Large $T(\Delta w; \mathbf{w})$ indicates that zeroing out Δw will lead to much utility cost to $\mathbf{w}$, thereby Δw should be preserved in $\Delta\mathbf{w}$. On the contrary, the updates that have little impact on model performance would be zeroed out. Suppose there are J layers in the model. Let $\mathbf{w}_j \in \mathbb{R}^{d_j}$ denote the weight in the j-th layer and let $T_s(\Delta\mathbf{w}_j)$ denote the s-th largest value of set $\{T(\Delta w; \mathbf{w}) \mid w \in \mathbf{w}_j\}$. To make local update $\Delta\mathbf{w}$ sparse, we define a mask function to generate 0-1 mask matrix for update Δw in the j-th layer of model $\mathbf{w}$ as

$$M_j(\Delta w; \mathbf{w}, s_j) \triangleq \begin{cases} 1, & \text{if } T(\Delta w; \mathbf{w}) \geq T_s(\Delta\mathbf{w}_j) \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

where $M_j(w; \mathbf{w}, s) \in \mathbb{R}^{d_j}$ is the mask matrix for layer update $\Delta\mathbf{w}_j$. Let $M(\Delta\mathbf{w}, s)$ denotes the mask matrix for model update $\Delta\mathbf{w}$, which is constructed by applying Eq. 12 to each layer. Then the sparsification process can be expressed as

$$\Delta\widetilde{\mathbf{w}} \leftarrow M(\Delta\mathbf{w}, s) \circ \Delta\mathbf{w} \quad (13)$$

where $\circ$ denotes Hadamard-product. After sparsification, for each layer update $\Delta\widetilde{\mathbf{w}}_j$, s_j update values from $\Delta\mathbf{w}_j$ that have largest $T(w; \mathbf{w})$ values are preserved and others are zeroed out. As a result, $\|\Delta\widetilde{\mathbf{w}}\|$ would be consistently smaller than $\|\Delta\mathbf{w}\|$. By adjusting s, we can control the sparsity of local update, thereby adjusting the norm reduction, to improve the utility of uploaded model updates.

5. Theoretical Results

In this section, we give the formal privacy guarantee and rigorous convergence analysis of our FL framework.

5.1. Privacy Analysis

In this subsection, we give the formal privacy guarantee. Same with DP-FedAvg, our method applies Gaussian mechanism to each agent's local update to ensure DP guarantee. At each communication round, the privacy guarantee of our method is equal to that of DP-FedAvg, if applying the same noise scale for both methods. For privacy cost accumulation, the composition theorem can be leveraged to compose the privacy cost at each round. In this paper, we make use of the moments accountant [1, 20] to obtain tighter privacy bounds than previous strong composition theorem [6].

Specifically, the moments accountant tracks a bound of the privacy loss random variable. Given a randomized mechanism $\mathcal{M}$, the privacy loss at output $o \in Range(\mathcal{M})$ is defined as $\ell(o; \mathcal{M}, \mathcal{D}, \mathcal{D}', \mathbf{aux}) \triangleq \log \frac{\Pr[\mathcal{M}(\mathcal{D}, \mathbf{aux}) = o]}{\Pr[\mathcal{M}(\mathcal{D}', \mathbf{aux}) = o]}$. Then, the privacy loss random variable $\mathcal{L}(o; \mathcal{M}, \mathcal{D}, \mathcal{D}', \mathbf{aux})$ is defined by evaluating the privacy loss at the outcome sampled from $\mathcal{M}(\mathcal{D})$. In our framework, the auxiliary information at round t is the current global weight $\mathbf{w}^{t-1}$. The moments accountant are defined as $\alpha_{\mathcal{M}}(\lambda) \triangleq \max_{\mathcal{D}, \mathcal{D}', \mathbf{aux}} \log \mathbb{E}[\exp(\lambda\mathcal{L}(\mathcal{M}, \mathcal{D}, \mathcal{D}', \mathbf{aux}))]$. According to the tail bound of moments accountant, $\mathcal{M}$ is (ϵ, δ)-DP with $\delta = \min_\lambda \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\varepsilon)$. Then, for an adaptive mechanism $\mathcal{M}_{1:K} = \mathcal{M}_1, \ldots, \mathcal{M}_K$, according to the composability of moments accountant, the privacy guarantee of $\mathcal{M}_{1:K}$ can be calculated by $\alpha_{\mathcal{M}_{1:K}}(\lambda) \leq \sum_{k=1}^{K} \alpha_{\mathcal{M}_k}(\lambda)$. Based on Theorem 1 in [1], we obtain the following theorem for privacy cost accumulation of FedAvg with our method Alg. 2 as local update method.

Theorem 1 (Privacy Guarantee). *Let P denote the number of participant clients in a communication round. There exist constants c_1 and c_2 so that given the number of communication rounds T, for any $\epsilon < c_1 q^2 T$, FedAvg that uses Alg. 2 as local update method satisfy (ϵ, δ) user-level DP for any $\delta > 0$, if we choose $\sigma \geq c_2 \frac{P\sqrt{T\log(1/\delta)}}{N\epsilon}$.*

5.2. Convergence Analysis

In this subsection, we present the convergence results of our method for general loss functions. Our analysis is based on the following assumptions:

Assumption 1 (L-Lipschitz Continuous Gradient). *There exists a constant $L > 0$, such that $\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, and $i \in \mathcal{P}$.*

Assumption 2 (Unbiased Local Gradient Estimator). *For any data sample z from $\mathcal{D}_i$, the local gradient estimator is unbiased, e.g., $\mathbb{E}[\nabla f_i(\mathbf{w}, z)] = \mathbb{E}[\nabla f_i(\mathbf{w})], \forall \mathbf{w} \in \mathbb{R}^d$ and $i \in \mathcal{P}$.*

Assumption 3 (Bounded Variance). *There exist two constants $\sigma_l>0$ and $\sigma_g>0$ such that for any $\mathbf{w} \in \mathbb{R}^d$ and $i \in \mathcal{P}$, the variance of each local gradient estimator is bounded by $\mathbb{E}\left[\left\|\nabla f_i\left(\mathbf{w}, z\right)-\nabla f_i\left(\mathbf{w}\right)\right\|^2\right] \leq \sigma_l^2$, for any data sample z from $\mathcal{D}_i$, and the global variance of the local gradient of the cost function is bounded by $\left\|\nabla f_i\left(\mathbf{w}\right)-\nabla f\left(\mathbf{w}\right)\right\|^2 \leq \sigma_g^2$.*

Assumption 4 (Bounded Gradient). *The loss function $f_i(\mathbf{w}; z)$ has G-bounded gradients, i.e., for any $\mathbf{w} \in \mathbb{R}^d$, $i \in \mathcal{P}$, and any data sample z from $\mathcal{D}_i$, we have $\|\nabla f_i(\mathbf{w}; z)\| \leq G$.*

Based on the above assumptions, we have the following convergence results:

Theorem 2 (Convergence of Our Protocol). *Under Assumptions 1-4, the sequence of outputs $\{\mathbf{w}^t\}$ generated by Alg. 1 with Alg. 2 as local update method satisfies:*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[\overline{\alpha}^t\left\|\nabla f\left(\mathbf{w}^t\right)\right\|^2\right] \leq \underbrace{\mathcal{O}\left(\frac{1}{\eta_g\eta_l QT}+\eta_l^2Q^2+\frac{\eta_g\eta_l}{P}\right)}_{\text{From FedAvg}}+\underbrace{\mathcal{O}\left(\frac{\eta_g\sigma^2S^2d}{\eta_l QP^2}\right)}_{\text{From operations for DP}}$$

where $\overline{\alpha}^t := \frac{1}{N}\sum_{i=1}^{N}\min\left(1, \frac{S}{\eta_l\beta_i^t\left\|\sum_{q=0}^{Q-1}\gamma_i^{t,q}\mathbf{g}_i^{t,q}\right\|}\right)$ with $\beta_i^t = \frac{\|M_i^t\circ\sum_{q=0}^{Q-1}\gamma_i^{t,q}\mathbf{g}_i^{t,q}\|}{\|\sum_{q=0}^{Q-1}\gamma_i^{t,q}\mathbf{g}_i^{t,q}\|}$.

The bound of Theorem 2, contains the first term inherited from standard FedAvg and the second term introduced by operations for DP guarantees. Comparing with the convergence rate of DP-FedAvg, our method achieves quadratic speedup convergence with respect to P in the second term, while that of DP-FedAvg is linear speedup [29]. To analyze the privacy/utility trade-off of our framework, we can replace the σ in Theorem 2 with that from Theorem 1. To analyze the impact of privacy parameters, let $S = \eta_l Qc$ with $c \geq G$ and σ^2 substituted. We can obtain the following results about privacy/utility trade-off.

Corollary 1 (Convergence with Privacy Guarantee). *Under Assumptions 1-4, for any clipping threshold $S \geq \eta_l QG$ and σ as in Theorem 1, for any (ϵ, δ) satisfying the constraints in Theorem 1, we have*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[\overline{\alpha}^t\left\|\nabla f\left(\mathbf{w}^t\right)\right\|^2\right] \leq \underbrace{\mathcal{O}\left(\frac{1}{\eta_g\eta_l QT}+\eta_l^2Q^2+\frac{\eta_g\eta_l}{P}\right)}_{\text{From FedAvg}}+\underbrace{\mathcal{O}\left(\frac{\eta_g\eta_l QTd\ln\left(\frac{1}{\delta}\right)}{N^2\epsilon^2}\right)}_{\text{From operations for DP}}$$

and the best rate one can obtain from the above bound is $\tilde{O}\left(\frac{\sqrt{d}}{N\epsilon}\right)$ by optimizing η_l, η_g, Q, T.

6. Experiment Settings

In this section, we conduct experiments to illustrate the advantages of DP-FedAvg with BLUR and LUS over the previous arts for FL with user-level DP guarantees.

Baselines. Our method aims at improving the performance of **DP-FedAvg** [18]. As a result, we choose DP-FedAvg as our baseline. DP-FedAvg ensures user-level DP guarantee by directly employing Gaussian mechanism to the local updates. To compare with SOTA methods, we also compare our method with previous works **DDGauss** [12] and **AE-DPFL** [30]. DDGauss ensures user-level DP by discretizing the data and adding discrete Gaussian noise before performing secure aggregation. AE-DPFL ensures user-level DP by a private voting mechanism with secure aggregation.

Datasets and models. We evaluate on two datasets: EMNIST and CIFAR-10. EMNIST is an image dataset with hand-written digits/letters over 62 classes grouped into 3400 clients by their writer. It substantially involves user-level DP with natural client heterogeneity and non-iid data distribution. CIFAR-10 is also an image dataset with 50K training samples and 10K testing samples over 10 classes. For CIFAR-10 dataset, we follow prior works [10, 31] to model non-iid data distributions using a Dirichlet distribution $\mathbf{Dir}(\alpha)$, in which a smaller α indicates higher data heterogeneity, as it makes the local distribution more biased. For both datasets, we conduct experiments on two models with different number of parameters: CNN-2-Layers model from [21] and ResNet-18 [9]. The model size is about 1.0M for CNN-2-Layers model and 11.1M for ResNet-18.

Configuration. For EMNIST and CIFAR-10 respectively, we set the number of rounds T to 1000 and 300, the default agent selection probability p to 0.04 and 0.06, the mini-batch size to 64 and 50, the local LR η_l to 0.03 and 0.1 For all experiments, the number of local iterations $Q = 30$, server LR $\eta_g = 1$. The privacy parameter $\delta = \frac{1}{N}$. For a specific ϵ, the clipping threshold S for vanilla DP-FedAvg is decided by grid search from $\{0.01, 0.03, 0.1, 0.3, 1.0\}$. We find $S = 0.03$ and $S = 0.3$ perform best on EMNIST and CIFAR-10, respectively. The hyper-parameter of BLUR is the regularization parameter λ. The hyper-parameter of LUS is the number of preserved updates s. Instead of using s, we define and adjust the *sparsity* $c = 1 - s/d$. A larger c indicates more update values are zerod out. While using BLUR and/or LUS, the hyper-parameters λ and c are chosen by grid search from $\{0.05, 0.1, 0.2, 0.4, 0.8\}$ and $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, respectively. The default λ and c are set to $\lambda = 0.4$ and $c = 0.7$.

Model	Setting	DP-FedAvg	AE-DPFL	DDGauss	Ours
CNN-2-Layers	$\epsilon = 2.0$	69.65 ± 0.74	71.16 ± 0.47	69.35 ± 0.61	$\mathbf{74.48 \pm 0.52}$
	$\epsilon = 4.0$	72.32 ± 0.81	74.63 ± 0.59	72.16 ± 0.76	$\mathbf{75.85 \pm 0.61}$
	$\epsilon = 6.0$	74.12 ± 0.75	76.25 ± 0.42	74.34 ± 0.70	$\mathbf{77.48 \pm 0.54}$
	$\epsilon = 8.0$	75.36 ± 0.64	77.41 ± 0.33	75.20 ± 0.68	$\mathbf{78.09 \pm 0.46}$
ResNet-18	$\epsilon = 2.0$	73.52 ± 0.53	76.37 ± 0.41	73.16 ± 0.58	$\mathbf{78.58 \pm 0.39}$
	$\epsilon = 4.0$	75.51 ± 0.60	79.22 ± 0.46	75.65 ± 0.64	$\mathbf{80.29 \pm 0.47}$
	$\epsilon = 6.0$	77.19 ± 0.55	80.24 ± 0.37	77.64 ± 0.61	$\mathbf{81.55 \pm 0.46}$
	$\epsilon = 8.0$	78.06 ± 0.49	81.33 ± 0.46	78.03 ± 0.52	$\mathbf{82.12 \pm 0.52}$

Table 1. Performance comparison under different privacy budgets on EMNIST dataset. A smaller ϵ indicates a stronger privacy guarantee.

Model	Setting	DP-FedAvg	AE-DPFL	DDGauss	Ours
CNN-2-Layers	$\alpha = 0.1$	53.84 ± 1.04	55.79 ± 0.86	53.55 ± 1.12	$\mathbf{58.95 \pm 0.95}$
	$\alpha = 1$	58.67 ± 0.85	60.00 ± 0.57	58.28 ± 0.96	$\mathbf{63.74 \pm 0.70}$
	$\alpha = 10$	62.25 ± 0.71	63.93 ± 0.45	62.43 ± 0.77	$\mathbf{65.34 \pm 0.52}$
	$\alpha = 100$	63.73 ± 0.64	64.51 ± 0.32	63.80 ± 0.69	$\mathbf{66.05 \pm 0.45}$
ResNet-18	$\alpha = 0.1$	59.73 ± 0.96	63.11 ± 0.65	59.37 ± 1.04	$\mathbf{64.50 \pm 0.88}$
	$\alpha = 1$	63.49 ± 0.81	65.80 ± 0.51	63.84 ± 0.89	$\mathbf{67.27 \pm 0.62}$
	$\alpha = 10$	65.64 ± 0.69	67.62 ± 0.42	65.85 ± 0.72	$\mathbf{68.96 \pm 0.54}$
	$\alpha = 100$	66.58 ± 0.60	68.39 ± 0.35	66.74 ± 0.63	$\mathbf{69.42 \pm 0.47}$

Table 2. Performance comparison given different data settings on CIFAR-10 dataset. A smaller α indicates higher data heterogeneity.

7. Experimental Results

Performance under different privacy budgets. Table 1 shows the test accuracies for different level privacy guarantees on EMNIST. Our method consistently outperforms the previous SOTA methods for private FL under different privacy budgets. Specifically, using BLUR and LUS can improve the accuracy of DP-FedAvg by $3\% \sim 4\%$ and $4\% \sim 5\%$ for CNN-2-Layers and ResNet-18, respectively. Comparing with SOTA methods, our method consistently provides significant improvements. For instance, on ResNet-18, our method provides gains of $4\% \sim 5\%$ to DDGauss and $1\% \sim 2\%$ to AE-DPFL. We also observe that the improvement on the larger model (ResNet-18) is relatively greater than that on the smaller model (CNN), which is a favorable advantage as we tend to use a large model to achieve better performance in practice. Moreover, the improvement for smaller ϵ is relatively greater than that for larger ϵ. For instance, the accuracy improvement over DP-FedAvg is 4.83% for $\epsilon = 2$, and 2.73% for $\epsilon = 8$ on CNN-2-Layers model. This is also a merit of our method as we tend to use smaller ϵ to ensure stronger DP guarantees.

Effectiveness of BLUR. We conduct experiments to validate the effectiveness of BLUR. The experiments are conducted on EMNIST with ResNet-18. The privacy budget is $\epsilon = 6.0$. To verify the effectiveness of BLUR, we study the performance of DP-FedAvg + BLUR with various regularization hyper-parameter λ from $\{0, 0.05, 0.1, 0.2, 0.4, 0.6, 0.8\}$, where $\lambda = 0$ indicates the vanilla DP-FedAvg. As shown in Figure 2, using BLUR consistently speeds up the convergence and improves the test accuracy of DP-FedAvg.

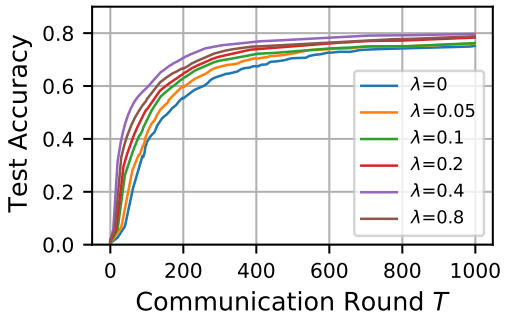

Figure 1. Effectiveness of BLUR with various λ. Vanilla DP-FedAvg is denoted by $\lambda = 0$. Using BLUR can consistently speed up the convergence and improve the test accuracy.

Method	Sparsity	Accuracy (%)	Gain (%)
DP-FedAvg	0.0	76.24	+0.00
DP-FedAvg + LUS	0.1	76.52	+0.28
	0.3	77.28	+1.04
	0.5	77.75	+1.51
	0.7	77.54	+1.30
	0.9	77.39	+1.15
DP-FedAvg + BLUR + LUS	0.1	78.28	+2.04
	0.3	79.26	+3.02
	0.5	79.97	+3.73
	0.7	80.32	+4.08
	0.9	80.17	+3.93

Table 3. Effectiveness of LUS with different sparsity. Using LUS consistently improves the accuracy and is synergistic with BLUR.

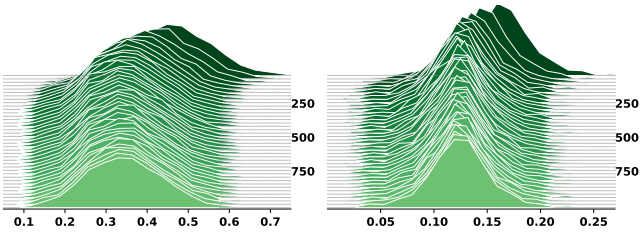

Figure 2. Distributions of local update norms (before clipping) at each round from DP-FedAvg (left) and ours (right). The y-axis and x-axis denote communication rounds and local update norms, respectively.

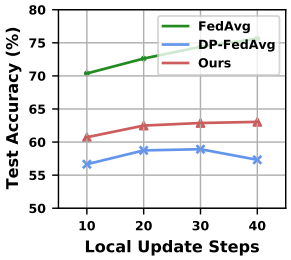

Figure 3. Impact of local update steps on CIFAR-10.

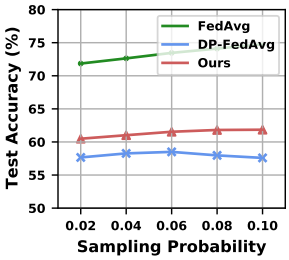

Figure 4. Impact of active agent numbers on CIFAR-10.

Effectiveness of LUS. To validate the effectiveness of LUS, we conduct experiments on DP-FedAvg + LUS and DP-FedAvg + BLUR + LUS with various sparsity c from $\{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$, where sparsity=0 indicates not using LUS. From Table 3, we observe that when equipping DP-FedAvg with LUR solely, the performance of DP-FedAvg are improved by about $0.58\% \sim 1.51\%$. However, compared with DP-FedAvg + BLUR, DP-FedAvg + BLUR + LUS obtains more performance gains by at most 2.07%, which indicates that the effectiveness of LUS can be boosted while cooperating with BLUR, and certifies that the effects of BLUR and LUS are synergistic.

Effects of bounding local update norms. To verify the effects of our method on bounding the norm of local updates, we show in Figure 2 the distributions of local updates norm before clipping in each communication round. The clipping bound is set to be 0.1 for both DP-FedAvg and our method. In contrast to DP-FedAvg, the clipping operation distorts less information in our framework, witnessed by a much smaller difference in the norm of local updates and the clipping threshold, which is smaller than 0.1 in most cases. Moreover, the local updates used in our method exhibit much less variance compared with DP-FedAvg. This is consistent with our motivation of making the local updates more adaptive to clipping by naturally reducing the norm of local updates before clipping.

Impacts of data heterogeneity. We explore different data heterogeneity by changing α for Dirichlet distribution in Table 2. We observe that our method consistently outperforms other baselines for different data heterogeneity. Moreover, using BLUR and LUS can lead to more accuracy gain when data heterogeneity is higher. For example, the accuracy gain is 5.11% for $\alpha = 0.1$, and 2.32% for $\alpha = 100$ on CNN-2-Layers. The reason for this could be that when data heterogeneity is higher, the local data distribution is more biased to the global distribution, leading to larger norm of local updates. Therefore, the clipped local updates are more biased to the original local updates. Employing BLUR and LUS can mitigate this by bounding the norm of local updates.

Impacts of communication frequency. We explore different local updating steps Q on CIFAR-10, so that a larger Q means longer communication delays before the global communication. Results in Figure 3 indicates that our approach is robust against different levels of communication delays while DP-FedAvg leads to performance degradation when Q is large, e.g. $Q = 40$. This is because that updating local models for more steps makes the updated local models more far away from the global model, leading to larger norm of local updates. On the contrary, our method can effectively limit the norm of local updates, thereby reducing the accuracy drop caused by clipping.

Impacts of active agents. We explore different agent sampling probability p on CIFAR-10. Using larger p means more agents participate in each round of communication but also requires more noise injection to the local updates according to Theorem 1. Results in Figure 4 indicates that equipping DP-FedAvg with BLUR and LUS makes it more robust against different levels of agent sampling rates.

8. Conclusion

We study the cause of model utility degradation in federated learning with DP and find the key is to naturally bound the local update norms before clipping. We then propose local regularization and sparsification methods to solve the problem. We provide theoretical analysis on the convergence and privacy of our framework. Experiments show that our framework significantly improves model utility over SOTA for federated learning with DP guarantee.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China (No. 2020AAA0103402), the Strategic Priority Research Program of Chinese Academy of Sciences (No. XDA27040300 and No. XDB32050200), the National Natural Science Foundation of China (No.62106267).

References

[1] Martín Abadi, Andy Chu, I. Goodfellow, H. B. McMahan, Ilya Mironov, Kunal Talwar, and L. Zhang. Deep learning with differential privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[2] D. A. Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, P. Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *ICLR*, 2021.

[3] Naman Agarwal, A. T. Suresh, F. Yu, Sanjiv Kumar, and H. B. McMahan. cpsgd: Communication-efficient and differentially-private distributed sgd. In *NeurIPS*, 2018.

[4] Nicholas Carlini, Chang Liu, Ú. Erlingsson, Jernej Kos, and D. Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX Security Symposium*, 2019.

[5] Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[6] C. Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9:211–407, 2014.

[7] Robin Geyer, T. Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *ArXiv*, abs/1712.07557, 2017.

[8] Song Han, Huizi Mao, and W. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *arXiv: Computer Vision and Pattern Recognition*, 2016.

[9] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

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

[11] Rui Hu, Yanmin Gong, and Yuanxiong Guo. Federated learning with sparsification-amplified privacy and adaptive optimization. In *IJCAI*, 2021.

[12] P. Kairouz, Ziyu Liu, and T. Steinke. The distributed discrete gaussian mechanism for federated learning with secure aggregation. *ArXiv*, abs/2102.06387, 2021.

[13] P. Kairouz, H. B. McMahan, B. Avent, Aurélien Bellet, M. Bennis, A. Bhagoji, Keith Bonawitz, Zachary B. Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, S. Rouayheb, David Evans, Josh Gardner, Zachary Garrett, A. Gascón, Badih Ghazi, Phillip B. Gibbons, M. Gruteser, Z. Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, T. Javidi, Gauri Joshi, M. Khodak, Jakub Konecný, Aleksandra Korolova, F. Koushanfar, O. Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, Mariana Raykova, Hang Qi, D. Ramage, R. Raskar, D. Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, A. T. Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, F. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14:1–210, 2021.

[14] Hao Li, Asim Kadav, Igor Durdanovic, H. Samet, and H. Graf. Pruning filters for efficient convnets. *ArXiv*, abs/1608.08710, 2017.

[15] Tian Li, Anit Kumar Sahu, Ameet S. Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37:50–60, 2020.

[16] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and W. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *ArXiv*, abs/1712.01887, 2018.

[17] H. B. McMahan, Eider Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

[18] H. B. McMahan, D. Ramage, Kunal Talwar, and L. Zhang. Learning differentially private recurrent language models. In *ICLR*, 2018.

[19] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Inference attacks against collaborative learning. *ArXiv*, abs/1805.04049, 2018.

[20] Ilya Mironov. Rényi differential privacy. *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275, 2017.

[21] Sashank J. Reddi, Zachary B. Charles, M. Zaheer, Zachary Garrett, Keith Rush, Jakub Konecný, Sanjiv Kumar, and H. B. McMahan. Adaptive federated optimization. *ArXiv*, abs/2003.00295, 2021.

[22] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, M. Zaheer, Ameet S. Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv: Learning*, 2020.

[23] Felix Sattler, Simon Wiedemann, K. Müller, and W. Samek. Sparse binary compression: Towards distributed deep learning with minimal communication. *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.

[24] R. Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2017.

[25] Lichao Sun and L. Lyu. Federated model distillation with noise-free differential privacy. In *IJCAI*, 2021.

[26] Lichao Sun, Jianwei Qian, Xun Chen, and Philip S. Yu. Ldp-fl: Practical private aggregation in federated learning with local differential privacy. In *IJCAI*, 2021.

[27] Om Thakkar, Galen Andrew, and H. B. McMahan. Differentially private learning with adaptive clipping. *ArXiv*, abs/1905.03871, 2019.

[28] Yusuke Tsuzuku, Hiroto Imachi, and Takuya Akiba. Variance-based gradient compression for efficient distributed deep learning. *ArXiv*, abs/1802.06058, 2018.

[29] Xinwei Zhang, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Jinfeng Yi. Understanding clipping for federated learning: Convergence and client-level differential privacy. *arXiv preprint arXiv:2106.13673*, 2021.

[30] Yuqing Zhu, Xiang Yu, Yi-Hsuan Tsai, F. Pittaluga, M. Faraki, Manmohan Chandraker, and Yu-Xiang Wang. Voting-based approaches for differentially private federated learning. *ArXiv*, abs/2010.04851, 2020.

[31] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. *arXiv preprint arXiv:2105.10056*, 2021.