SoteriaFL: A Unified Framework for Private Federated Learning with Communication Compression

Anonymous Author(s)
Affiliation
Address
email

Abstract

To enable large-scale machine learning in bandwidth-hungry environments such as wireless networks, significant progress has been made recently in designing communication-efficient federated learning algorithms with the aid of communication compression. On the other end, privacy-preserving, especially at the client level, is another important desideratum that has not been addressed simultaneously in the presence of advanced communication compression techniques yet. In this paper, we propose a unified framework that enhances the communication efficiency of private federated learning with communication compression. Exploiting both general compression operators and local differential privacy, we first examine a simple algorithm that applies compression directly to differentially-private stochastic gradient descent, and identify its limitations. We then propose a unified framework SoteriaFL for private federated learning, which accommodates a general family of local gradient estimators including popular stochastic variance-reduced gradient methods and the state-of-the-art shifted compression scheme. We provide a comprehensive characterization of its performance trade-offs in terms of privacy, utility, and communication complexity, where SoteriaFL is shown to achieve better communication complexity without sacrificing privacy nor utility than other private federated learning algorithms without communication compression.

1 Introduction

With the proliferation of mobile and edge devices, federated learning (FL) [37, 44] has recently emerged as a disruptive paradigm for training large-scale machine learning models over a vast amount of geographically distributed and heterogeneous devices. For instance, Google uses FL in the Gboard mobile keyboard for next word predictions [25]. FL is often modeled as a distributed optimization problem [36, 37, 44, 31, 56], aiming to solve

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\left\{f(\boldsymbol{x};D):=\frac{1}{n}\sum_{i=1}^{n}f(\boldsymbol{x};D_i)\right\},\quad \text{where } f(\boldsymbol{x};D_i):=\frac{1}{m}\sum_{j=1}^{m}f(\boldsymbol{x};d_{i,j}). \tag{1}$$

Here, D denotes the entire dataset distributed across all n clients, where each client i has a local dataset $D_i=\{d_{i,j}\}_{j=1}^m$ of equal size m,[1] $\boldsymbol{x}\in\mathbb{R}^d$ denotes the model parameters, $f(\boldsymbol{x};D)$, $f(\boldsymbol{x};D_i)$, and $f(\boldsymbol{x};d_{i,j})$ denote the nonconvex loss function of the current model $\boldsymbol{x}$ on the entire dataset D, the local dataset D_i, and a single data sample $d_{i,j}$, respectively. For simplicity, we use $f(\boldsymbol{x})$, $f_i(\boldsymbol{x})$ and $f_{i,j}(\boldsymbol{x})$ to denote $f(\boldsymbol{x};D)$, $f(\boldsymbol{x};D_i)$ and $f(\boldsymbol{x};d_{i,j})$, respectively.

[1] This is without loss of generality, since otherwise one can simply adjust the weights of the loss function.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

1.1 Motivation: privacy-utility-communication trade-offs

To unleash the full potential of FL, it is extremely important that the algorithm designed to solve (1) needs to meet several competing desiderata.

Communication efficiency Communication between the server and clients is well recognized as the main bottleneck for optimizing the latency of FL systems, especially when the clients—such as mobile devices—have limited bandwidth, the number of clients is large, and/or the machine learning model has a lot of parameters—for example, the language model GPT-3 [6] has billions of parameters and therefore cumbersome to share directly.

Therefore, it is very important to design FL algorithms to reduce the overall communication cost, which takes into account both *the number of communication rounds* and *the communication cost per communication round* for reaching a desired accuracy. With these two quantities in mind, there are two principal approaches for communication-efficient FL: 1) *local methods*, where in each communication round, clients run multiple local update steps before communicating with the server, in the hope of reducing the number of communication rounds, e.g., FedAvg [44], Local-SVRG [23], SCAFFOLD [34], FedPAGE [61], and ProxSkip [46]; 2) *compression methods*, where clients send compressed communication message to the server, in the hope of reducing the communication cost per communication round, e.g., [3, 35, 55, 27, 33, 45, 47, 24, 41, 48, 40]. While both categories have garnered significant attention in recent years, we will focus on the second approach based on communication compression to enhance communication efficiency.

Privacy preserving While FL holds great promise of harnessing the inferential power of private data stored on a large number of distributed clients, these local data at clients often contain sensitive or proprietary information without consent to share. Although FL may appear to protect the data privacy via storing data locally and only sharing the model updates (e.g., gradient information), the training process can nonetheless reveal sensitive information as demonstrated by, e.g., Zhu et al. [63]. It is thus desirable for FL to preserve privacy in a guaranteed manner [21, 31, 49, 56].

To ensure the training process does not accidentally leak private information, advanced privacy-preserving tools such as *differential privacy* (DP) [18] have been widely integrated into training algorithms [16, 10, 17, 1, 54, 28, 13, 20]. A notable example is DP-SGD [1], which developed a differentially-private stochastic gradient descent (SGD) algorithm in the centralized (single-node) setting. More recently, several differentially-private algorithms [29, 57, 50, 43] are proposed for the more general distributed (n-node) setting suitable for FL. In this paper, we also follow the DP approach to preserve privacy. In particular, we adopt local differential privacy (LDP) to respect the privacy of each client, which is critical in FL.

Encouraged by recent advances in communication compression techniques, and the widespread success of differentially-private methods, a natural question is

Can we develop a unified framework for private federated learning with communication compression, and understand the trade-offs between privacy, utility, and communication?

Note that there have been a handful of works that simultaneously address compression and privacy in FL. Unfortunately, they only provide partial answers to the above question. Most of the existing works only consider specific, elementary or tailored compression schemes that are applied directly to the gradient messages in DP-SGD [2, 58, 22, 64, 60, 15]. A number of works [51, 11, 12, 32, 19, 52] extended and considered different compression schemes, but did not provide concrete trade-offs in terms of privacy, utility and communication. Furthermore, existing theoretical analyses can be limited only to convex problems [22], lacking in some aspects such as utility [64], or delivering pessimistic guarantees on utility and / or communication due to strong assumptions [60, 15]. Finally, existing work only studied the DP framework for direct compression, while it is known that the recently developed shifted compression scheme [45, 26] achieves much better convergence. Due to noise injection for privacy-preserving, it is a priori unclear if the shifted compression scheme is also compatible with privacy.

1.2 Our contributions

In this paper, we answer the above question by providing a general approach that enhances the communication efficiency of private federated learning in the *nonconvex* setting, through a unified framework called SoteriaFL (see Algorithm 2). Specifically, we have the following contributions.

Table 1: Comparisons among (local) differentially-private algorithms for the nonconvex problem (1) in both central (single-node) and distributed (n-node) settings. Here, m denotes the number of data stored on a single client, n is the number of clients, d is the dimension, and ω is the parameter for the compression operator (cf. Definition 1). The communication complexity is computed by $ndT/(1+\omega)$, where T is the total number of communication rounds, and $nd/(1+\omega)$ is the communication cost per round. The utility / accuracy measures the average squared gradient norm of the objective function after T rounds. Note that the algorithm is better when the utility / accuracy and the communication complexity are small under the same privacy guarantee.

Algorithm	**Privacy**	**Utility / Accuracy**	**Communication Complexity**	**Remark**
RPPSGD [59]	(ϵ, δ)-DP	$\frac{\sqrt{d\log(m/\delta)\log(1/\delta)}}{m\epsilon}$	—	single node
DP-GD/SGD [1, 54]	(ϵ, δ)-DP	$\frac{\sqrt{d\log(1/\delta)}}{m\epsilon}$	—	single node
DP-SRM [57]	(ϵ, δ)-DP	$\frac{\sqrt{d\log(1/\delta)}}{m\epsilon}$	—	single node
Distributed DP-SRM [57] (1)	(ϵ, δ)-DP	$\frac{\sqrt{d\log(1/\delta)}}{nm\epsilon}$	$\frac{n^2 m\epsilon\sqrt{d}}{\sqrt{\log(1/\delta)}}$	n nodes, no comp.
LDP SVRG [43]	(ϵ, δ)-LDP	$\frac{\sqrt{d\log(1/\delta)}}{\sqrt{n}m\epsilon}$	$\frac{n^{3/2} m\epsilon\sqrt{d}}{\sqrt{\log(1/\delta)}}$	n nodes, no comp.
LDP SPIDER [43]	(ϵ, δ)-LDP	$\frac{\sqrt{d\log(1/\delta)}}{\sqrt{n}m\epsilon}$	$\frac{n^{3/2} m\epsilon\sqrt{d}}{\sqrt{\log(1/\delta)}}$	n nodes, no comp.
Q-DPSGD-1 [15] (2)	(ϵ, δ)-LDP	$\frac{\tilde{\sigma}^2\epsilon^{2/3}}{n} + \frac{d\log(1/\delta)}{m^2\epsilon^{10/3}} + \frac{\epsilon^{2/3}}{m}$	$\frac{nd}{\epsilon^2\tilde{\sigma}^2}$	n nodes, direct comp.
SDM-DSGD [60] (3)	(ϵ, δ)-LDP	$\tilde{O}\left(\frac{\sqrt{d\log(1/\delta)}}{\sqrt{n}m\epsilon}\right)$	$\frac{n^{7/2} m\epsilon\sqrt{d}}{(1+\omega)^{3/2}\sqrt{\log(1/\delta)}} + \frac{nm^2\epsilon^2}{(1+\omega)\log(1/\delta)}$	n nodes, direct comp.
CDP-SGD (Theorem 1)	(ϵ, δ)-LDP	$\frac{\sqrt{(1+\omega)d\log(1/\delta)}}{\sqrt{n}m\epsilon}$	$\frac{n^{3/2} m\epsilon\sqrt{d}}{(1+\omega)^{3/2}\sqrt{\log(1/\delta)}} + \frac{nm^2\epsilon^2}{(1+\omega)\log(1/\delta)}$	n nodes, direct comp.
SoteriaFL-GD SoteriaFL-SGD (4) (Corollary 1)	(ϵ, δ)-LDP	$\frac{\sqrt{(1+\omega)d\log(1/\delta)}}{\sqrt{n}m\epsilon}(1+\sqrt{\tau})$	$\frac{n^{3/2} m\epsilon\sqrt{d}}{(1+\omega)^{3/2}\sqrt{\log(1/\delta)}}(1+\sqrt{\tau})$	n nodes, shifted comp.
SoteriaFL-SVRG SoteriaFL-SAGA (4) (Corollary 2)	(ϵ, δ)-LDP	$\frac{\sqrt{(1+\omega)d\log(1/\delta)}}{\sqrt{n}m\epsilon}$	$\frac{n^{3/2} m\epsilon\sqrt{d}}{(1+\omega)^{3/2}\sqrt{\log(1/\delta)}}(1+\tau)$	n nodes, shifted comp.

(1) Wang et al. [57] considered the "global" (ϵ, δ)-DP (which only protects the privacy for entire dataset D, i.e., the local dataset D_i on node i may leak to other nodes $j \neq i$) without communication compression. However, we consider the "local" (ϵ, δ)-LDP which can protect the local datasets D_i's at the client level.

(2) Ding et al. [15] adopted a slightly different compression assumption $\mathbb{E}[\|\mathcal{C}(\boldsymbol{x}) - \boldsymbol{x}\|^2] \leq \tilde{\sigma}^2$, with $\tilde{\sigma}^2$ playing a similar role as $(1+\omega)$ in ours. It only works for small $\epsilon = 1/\sqrt{T}$, which leads to an accuracy no better than $\frac{\tilde{\sigma}^2\epsilon^{2/3}}{n} + \frac{d\log(1/\delta)}{m^2\epsilon^{10/3}} \geq 2\frac{\sqrt{\tilde{\sigma}^2 d\log(1/\delta)}}{\sqrt{n}m\epsilon} \cdot \epsilon^{-1/3} \overset{\epsilon=1/\sqrt{T}}{=} 2\frac{\sqrt{\tilde{\sigma}^2 d\log(1/\delta)}}{\sqrt{n}m\epsilon} \cdot T^{1/6}$, a factor of $T^{1/6}$ worse than the utility of the other algorithms including ours.

(3) Zhang et al. [60] only considered random-k sparsification, which is a special case of our general compression operator. Moreover, it requires $1+\omega \ll \log T$, i.e., at least $k \gg \frac{d}{\log T}$ out of d coordinates need to be communicated, and its utility hides logarithmic factors larger than $1+\omega$. The communication complexity $n^{7/2}$ is due to their convergence condition $T > n^5$.

(4) Here, $\tau := \frac{(1+\omega)^{3/2}}{n^{1/2}}$. If $n \geq (1+\omega)^3$ (which is typical in FL), then $\tau < 1$, and we can drop the terms involving τ from SoteriaFL.

1. We first present a simple CDP-SGD (Algorithm 1) that directly combines communication compression and DP-SGD. We provide theoretical analysis for CDP-SGD in Theorem 1 and show its limitations in communication efficiency.
2. We then propose a general framework SoteriaFL for private FL, which accommodates a general family of local gradient estimators including popular stochastic variance-reduced gradient methods and the state-of-the-art shifted compression scheme. We provide a unified characterization of its performance trade-offs in terms of privacy, utility (convergence accuracy), and communication complexity.
3. We apply our unified analysis for SoteriaFL and obtain theoretical guarantees for several new private FL algorithms, including SoteriaFL-GD, SoteriaFL-SGD, SoteriaFL-SVRG, and SoteriaFL-SAGA. All of these algorithms are shown to perform better than the plain CDP-SGD (Algorithm 1), and have lower communication complexity compared with other private FL algorithms without compression.

We provide detailed comparisons between the proposed approach and prior arts in Table 1. To the best of our knowledge, SoteriaFL is the first unified framework that simultaneously enables local differential privacy and shifted compression, and allows flexible local computation protocols at the client level.

2 Preliminaries

Let $[n]$ denote the set $\{1, 2, \cdots, n\}$ and $\|\cdot\|$ denote the Euclidean norm of a vector and the spectral norm of a matrix. Let $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$ denote the standard Euclidean inner product of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. Let $f^* := \min_{\boldsymbol{x}} f(x)$ denote the optimal value of problem (1). In addition, we use the standard order notation $O(\cdot)$ to hide absolute constants. We now introduce the definitions of the compression operator and local differential privacy, and some assumptions for the objective functions.

Compression operator We introduce the notion of a randomized *compression operator*, which is used to compress the gradients to save communication. The following definition of unbiased compressors is standard and has been used in many distributed/federated learning algorithms [3, 35, 45, 26, 42, 24, 41].

Definition 1 (Compression operator)**.** *A randomized map $\mathcal{C} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is an ω-compression operator if for all $\boldsymbol{x} \in \mathbb{R}^d$, it satisfies*

$$\mathbb{E}[\mathcal{C}(\boldsymbol{x})] = \boldsymbol{x}, \qquad \mathbb{E}\left[\|\mathcal{C}(\boldsymbol{x}) - \boldsymbol{x}\|^2\right] \leq \omega \|\boldsymbol{x}\|^2 . \tag{2}$$

In particular, no compression ($\mathcal{C}(\boldsymbol{x}) \equiv \boldsymbol{x}$) implies $\omega = 0$.

Note that the conditions (2) are satisfied by many practically useful compression operators, e.g., random sparsification and random quantization [3, 42, 41]. A useful rule of thumb is that the communication cost is often reduced by a factor of $\frac{1}{1+\omega}$ due to compression [3]. Next, we briefly discuss an example called random sparsification to provide more intuition.

Example 1 (Random sparsification). Given $\boldsymbol{x} \in \mathbb{R}^d$, the random-$k$ sparsification operator is defined by $\mathcal{C}(\boldsymbol{x}) := \frac{d}{k} \cdot (\boldsymbol{\xi}_k \odot \boldsymbol{x})$, where $\odot$ denotes the Hadamard (element-wise) product and $\boldsymbol{\xi}_k \in \{0, 1\}^d$ is a uniformly random binary vector with k nonzero entries ($\|\boldsymbol{\xi}_k\|_0 = k$). This random-$k$ sparsification operator $\mathcal{C}$ satisfies (2) with $\omega = \frac{d}{k} - 1$, and the communication cost is reduced by a factor of $\frac{1}{1+\omega}$ since we transmit $k = \frac{d}{1+\omega}$ (due to its $\omega = \frac{d}{k} - 1$) coordinates rather than d coordinates.

Local differential privacy We not only want to train the machine learning model using fewer communication bits, but also want to maintain each client's local privacy, which is a key component for FL applications. We follow the framework of (local) differential privacy [4, 9, 62]. We say that two datasets D and D' are neighbors if they differ by only one entry. We have the following definition for local differential privacy (LDP).

Definition 2 (Local differential privacy (LDP))**.** *A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ is (ϵ, δ)-locally differentially private for client i if for all neighboring datasets $D_i, D_i' \in \mathcal{D}$ on client i and for all events $S \in \mathcal{R}$ in the output space of $\mathcal{M}$, we have*

$$\Pr\{\mathcal{M}(D_i) \in S\} \leq e^{\epsilon} \Pr\{\mathcal{M}(D_i') \in S\} + \delta.$$

The definition of LDP (Definition 2) is very similar to the original definition of (ϵ, δ)-DP [18, 17], except that now in the FL setting, each client protects its own privacy by encoding and processing its sensitive data locally, and then transmitting the encoded information to the server without coordination and information sharing between the clients.

Assumptions about the functions Recalling (1), we consider the *nonconvex* FL setting, where the functions $\{f_{i,j}\}$ are arbitrary functions satisfying the following standard smoothness assumption (Assumption 1) and bounded gradient assumption (Assumption 2).

Assumption 1 (Smoothness)**.** *There exists some $L \geq 0$, such that for all $i \in [n], j \in [m]$, the function $f_{i,j}$ is L-smooth,*

$$\|\nabla f_{i,j}(\boldsymbol{x}_1) - \nabla f_{i,j}(\boldsymbol{x}_2)\| \leq L \|\boldsymbol{x}_1 - \boldsymbol{x}_2\| , \qquad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d.$$

Assumption 2 (Bounded gradient)**.** *There exists some $G \geq 0$, such that for all $i \in [n], j \in [m]$ and $\boldsymbol{x} \in \mathbb{R}^d$, we have $\|\nabla f_{i,j}(\boldsymbol{x})\| \leq G$.*

The smoothness assumption is very standard for the convergence analysis, and the bounded gradient assumption is also standard for the differential privacy analysis [5, 54, 28, 20].

3 Warm-up: Plain Compressed Differentially-Private SGD

There are two methods to combine privacy and compression: (1) first perturb and then compress, and (2) first compress and then perturb. The advantage of the first method is that it is very simple and general, since compression will preserve the differential privacy and work seamlessly with any previous existing privacy mechanisms. However, the second method needs to design careful perturbation mechanisms (otherwise the perturbation might diminish the communication saving of compression), e.g., binomial perturbation [2] or discrete Gaussian perturbation [32]. In addition, it is observed that the first method achieves better utility compared with the second one in some settings [15]. Thus, we also apply the first method in this paper: first perturb then compress.

As a warm-up, we first introduce a simple algorithm CDP-SGD (described in Algorithm 1), which subsumes some existing algorithms as special cases (e.g., [60, 15, 64]) for private FL with better theoretical guarantees. The procedure for CDP-SGD is very simple: at each round t, each client first computes a local stochastic gradient $\tilde{\boldsymbol{g}}_i^t$ using its local dataset D_i (Line 4 in Algorithm 1). Then, it uses Gaussian mechanism [1] to achieve LDP (Line 5 in Algorithm 1) and transfers the compressed perturbed private gradient information to the server (Line 6 in Algorithm 1). Finally, the server aggregates the compressed information and update the model parameters (Line 8-9 in Algorithm 1).

Algorithm 1 Compressed Differentially-Private Stochastic Gradient Descent (CDP-SGD)

Input: initial point x^0, stepsize η_t, variance σ_p^2, minibatch size b

1: **for** $t = 0, 1, 2, \ldots, T$ **do**
2: **for each node** $i \in [n]$ **do in parallel**
3: Sample a random minibatch $\mathcal{I}_b$ from local dataset D_i
4: Compute local stochastic gradient $\tilde{\boldsymbol{g}}_i^t = \frac{1}{b}\sum_{j \in \mathcal{I}_b} \nabla f_{i,j}(\boldsymbol{x}^t)$ // all nodes use SGD method
5: *Privacy*: $\boldsymbol{g}_i^t = \tilde{\boldsymbol{g}}_i^t + \boldsymbol{\xi}_i^t$, where $\boldsymbol{\xi}_t^i \sim \mathcal{N}(\boldsymbol{0}, \sigma_p^2 \boldsymbol{I})$
6: *Compression*: let $\boldsymbol{v}_i^t = \mathcal{C}_i^t(\boldsymbol{g}_i^t)$ and send to the server // direct compression
7: **end each node**
8: Server aggregates compressed information $\boldsymbol{v}^t = \frac{1}{n}\sum_{i=1}^n \boldsymbol{v}_i^t$
9: $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta_t \boldsymbol{v}^t$
10: **end for**

Now we present the theoretical guarantees for CDP-SGD in the following theorem.

Theorem 1 (Utility and communication for CDP-SGD)**.** *Suppose that Assumptions 1 and 2 hold. By choosing the algorithm parameters properly and letting the total number of communication rounds $T = O\Big(\frac{\sqrt{nL}m\epsilon}{G\sqrt{(1+\omega)d\log(1/\delta)}} + \frac{m^2\epsilon^2}{d\log(1/\delta)}\Big)$, CDP-SGD (Algorithm 1) satisfies (ϵ, δ)-LDP and the utility $\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{x}_t)\|^2 \leq O\Big(\frac{G\sqrt{(1+\omega)Ld\log(1/\delta)}}{\sqrt{n}m\epsilon}\Big)$.*

The proposed CDP-SGD (Algorithm 1) is simple but effective. When the compression parameter ω is a constant (i.e., constant compression ratio), CDP-SGD achieves the same utility $O\Big(\frac{\sqrt{d\log(1/\delta)}}{m\epsilon}\Big)$ as DP-SGD in the single-node case $n = 1$. Our utility is better than [15] by a factor of $T^{1/6}$, and our communication complexity is much better than [60] (see Table 1).

However, the communication complexity of CDP-SGD still has room for improvements due to *direct compression* (Line 6 in Algorithm 1). In particular, if the local dataset size m stored on clients is dominating, then CDP-SGD (even if we compute local full gradients as CDP-GD) requires $O(m^2)$ communication rounds (see Theorem 1), while previous distributed differentially-private algorithms without communication compression (Distributed DP-SRM [57], LDP SVRG and LDP SPIDER [43]) only need $O(m)$ communication rounds (see Table 1).

4 SoteriaFL: Unified Private FL Framework with Shifted Compression

Due to the limitations of plain CDP-SGD, we now present an advanced and unified private FL framework called SoteriaFL in this section, which allows a large family of local gradient estimators (Line 4 in Algorithm 2 and Line 3–11 in Algorithm 3). Via adopting the advanced *shifted compression* (Line 6 in Algorithm 2), SoteriaFL reduces the total number of communication rounds $O(m^2)$ of CDP-SGD to $O(m)$, which matches previous uncompressed DP algorithms (see Table 1), and further reduces the total communication complexity due to less communication cost per round.

We first introduce our unified SoteriaFL framework in Section 4.1 and then provide a generic assumption and a unified analysis that covers privacy, utility, and communication for SoteriaFL in Section 4.2.

4.1 Unified SoteriaFL framework

Our SoteriaFL framework is described in Algorithm 2. At each round t, each client will compute a local (stochastic) gradient estimator $\tilde{\boldsymbol{g}}_i^t$ using its local dataset D_i (Line 4 in Algorithm 2). One can choose several optimization methods for computing this local gradient estimator such as standard gradient descent (GD), stochastic GD (SGD), stochastic variance reduced gradient (SVRG) [30, 38], and SAGA [14] (see e.g., Line 3–11 in Algorithm 3). Then, each client adds a Gaussian perturbation $\boldsymbol{\xi}_i^t$ on its gradient estimate $\tilde{\boldsymbol{g}}_i^t$ to provide LDP (Line 5 in Algorithm 2). However, different from CDP-SGD (Algorithm 1) where we directly compress the perturbed stochastic gradients, now each client maintains a reference $\boldsymbol{s}_i^t$ and compress the shifted message $\tilde{\boldsymbol{g}}_i^t - \boldsymbol{s}_i^t$ (Line 6 in Algorithm 2). This extra shift operation achieves much better convergence behavior (fewer communication rounds) than CDP-SGD, and thus can achieve lower communication complexity.

Algorithm 2 SoteriaFL (a unified framework for compressed private FL)

Input: initial point x^0, stepsize η_t, shift stepsize γ_t, variance σ_p^2

1: Initialize shift $\boldsymbol{s}_i^0 = 0,\ \forall i \in [n]$
2: **for** $t = 0, 1, 2, \ldots, T$ **do**
3: **for each node** $i \in [n]$ **do in parallel**
4: Compute local gradient estimator $\tilde{\boldsymbol{g}}_i^t$ // it allows many methods, e.g., SGD, SVRG, and SAGA
5: *Privacy*: $\boldsymbol{g}_i^t = \tilde{\boldsymbol{g}}_i^t + \boldsymbol{\xi}_i^t$, where $\boldsymbol{\xi}_i^t \sim \mathcal{N}(\boldsymbol{0}, \sigma_p^2 \boldsymbol{I})$
6: *Compression*: let $\boldsymbol{v}_i^t = \mathcal{C}_i^t(\boldsymbol{g}_i^t - \boldsymbol{s}_i^t)$ and send to the server // shifted compression
7: Update shift $\boldsymbol{s}_i^{t+1} = \boldsymbol{s}_i^t + \gamma_t \mathcal{C}_i^t(\boldsymbol{g}_i^t - \boldsymbol{s}_i^t)$
8: **end each node**
9: Server aggregates compressed information $\boldsymbol{v}^t = \boldsymbol{s}^t + \frac{1}{n}\sum_{i=1}^n \boldsymbol{v}_i^t$
10: $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta_t \boldsymbol{v}^t$
11: $\boldsymbol{s}^{t+1} = \boldsymbol{s}^t + \gamma_t \frac{1}{n}\sum_{i=1}^n \boldsymbol{v}_i^t$
12: **end for**

4.2 Generic assumption and unified theory

We provide a generic Assumption 3 which is very flexible to capture the behavior of several existing (and potentially new) gradient estimators, while simultaneously maintaining the tractability to enable a unified and sharp theoretical analysis.

Assumption 3 (Generic assumption of local gradient estimator for SoteriaFL)**.** *The gradient estimator $\tilde{\boldsymbol{g}}_i^t$ (Line 4 of Algorithm 2) is unbiased $\mathbb{E}_t[\tilde{\boldsymbol{g}}_i^t] = \nabla f_i(\boldsymbol{x}^t)$ for $i \in [n]$, where $\mathbb{E}_t$ takes the expectation conditioned on all history before round t. Moreover, it can be decomposed as two terms $\tilde{\boldsymbol{g}}_i^t := \mathcal{A}_i^t + \mathcal{B}_i^t$ and there exist constants $G_A, G_B, C_1, C_2, C_3, C_4, \theta$ and a random sequence $\{\Delta^t\}$ such that*

$$\mathcal{A}_i^t = \frac{1}{b}\sum_{j\in\mathcal{I}_b} \varphi_{i,j}^t, \qquad \mathcal{B}_i^t = \frac{1}{m}\sum_{j=1}^m \psi_{i,j}^t, \tag{3a}$$

$$\mathbb{E}_t\Big[\frac{1}{n}\sum_{i=1}^n \|\tilde{\boldsymbol{g}}_i^t - \nabla f_i(\boldsymbol{x}^t)\|^2\Big] \le C_1\Delta^t + C_2, \tag{3b}$$

$$\mathbb{E}_t\left[\Delta^{t+1}\right] \le (1-\theta)\Delta^t + C_3\|\nabla f(\boldsymbol{x}^t)\|^2 + C_4\mathbb{E}_t\|\boldsymbol{x}^{t+1} - \boldsymbol{x}^t\|^2, \tag{3c}$$

where $\varphi_{i,j}^t$ and $\psi_{i,j}^t$ are bounded by G_A and G_B respectively, and $\mathcal{I}_b$ usually denotes a random minibatch with size b. Here, $\varphi_{i,j}^t$ and $\psi_{i,j}^t$ should be viewed as functions related to the j-th sample $d_{i,j}$ stored on client i.

A few comments are in order. Concretely, the decomposition (3a) is used for our unified privacy analysis (i.e., Theorem 2). We can let one of them be $\mathbf{0}$ if the gradient estimator only contains one term or is not decomposable. The parameters C_1 and C_2 in (3b) capture the variance of the gradient estimators, e.g., $C_1 = C_2 = 0$ if the client computes local full gradient $\tilde{\boldsymbol{g}}_i^t = \nabla f_i(\boldsymbol{x}^t)$, and $C_1 \neq 0$ (note that Δ^t will shrink in (3c)) and $C_2 = 0$ if the client uses variance-reduced gradient estimators such as SVRG / SAGA. Finally, the parameters θ, C_3 and C_4 in (3c) capture the shrinking behavior of the variance (incurred by the gradient estimators), where different variance-reduced gradient methods usually have different shrinking behaviors. More concrete examples to follow in Lemma 1 in Section 5.

Unified theory for privacy-utility-communication trade-offs Given our generic Assumption 3, we can obtain a unified analysis for SoteriaFL framework. The following Theorem 2 unifies the privacy analysis and Theorem 3 unifies the utility and communication complexity analysis.

Theorem 2 (Privacy for SoteriaFL)**.** *Suppose that Assumption 3 holds. There exist constants c and c', for any $\epsilon < c' b^2 T/m^2$ and $\delta \in (0,1)$,* SoteriaFL *(Algorithm 2) is (ϵ, δ)-LDP if we choose*

$$\sigma_p^2 = c\frac{(G_A^2 + G_B^2)T\log(1/\delta)}{m^2\epsilon^2}. \tag{4}$$

Theorem 3 (Utility and communication for SoteriaFL)**.** *Suppose that Assumptions 1 and 3 hold, and the compression operator $\mathcal{C}_i^t$ (used in Line 6 of Algorithm 2) satisfies (2). Set the stepsize as*

$$\eta_t \equiv \eta \leq \min\left\{\frac{1}{(1 + 2\alpha C_4 + 2\beta(1+2\omega) + 2\alpha C_3/\eta^2)L}, \frac{\sqrt{\beta n}}{\sqrt{1 + 2\alpha C_4 + 2\beta(1+2\omega)}(1+\omega)L}\right\},$$

where $\alpha = \frac{3\beta C_1}{2(1+\omega)\theta L^2}$, $\forall \beta > 0$, the shift stepsize as $\gamma_t \equiv \frac{1}{1+\omega}$, and the privacy variance σ_p^2 according to Theorem 2. Then, SoteriaFL *(Algorithm 2) satisfies (ϵ, δ)-LDP and the following*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{x}^t)\|^2 \leq \frac{2\Phi_0}{\eta T} + \frac{3\beta}{(1+\omega)L\eta}\left(C_2 + \frac{c(G_A^2 + G_B^2)dT\log(1/\delta)}{m^2\epsilon^2}\right),$$

where $\Phi_0 := f(x^0) - f^* + \alpha L \Delta^0 + \frac{\beta}{Ln}\sum_{i=1}^n \|\nabla f_i(x^0) - s_i^0\|^2$. By further choosing the total number of communication rounds T as

$$T = \max\left\{\frac{m\epsilon\sqrt{2(1+\omega)L\Phi_0}}{\sqrt{3\beta c d(G_A^2 + G_B^2)\log(1/\delta)}}, \frac{C_2 m^2\epsilon^2}{cd(G_A^2 + G_B^2)\log(1/\delta)}\right\}, \tag{5}$$

SoteriaFL *has the following utility (accuracy) guarantee:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{x}^t)\|^2 \leq O\left(\max\left\{\frac{\sqrt{\beta d(G_A^2 + G_B^2)\log(1/\delta)}}{\eta m\epsilon\sqrt{(1+\omega)L}}, \frac{\beta C_2}{(1+\omega)L\eta}\right\}\right). \tag{6}$$

Theorem 3 is a unified theorem for our SoteriaFL framework, which covers a large family of local stochastic gradient methods under the generic Assumption 3. In the next Section 5, we will show that many popular local gradient estimators (GD, SGD, SVRG, and SAGA) satisfy Assumption 3, and thus can be captured by our unified analysis.

5 Some Algorithms within SoteriaFL Framework

In this section, we propose several new algorithms (SoteriaFL-GD, SoteriaFL-SGD, SoteriaFL-SVRG and SoteriaFL-SAGA) captured by our SoteriaFL framework. We give a detailed Algorithm 3 which describes all these four SoteriaFL-type algorithms in a nutshell.

To begin, we show that these local gradient estimators (GD, SGD, SVRG, and SAGA) satisfy Assumption 3 in the following main lemma, by detailing the corresponding parameter values (i.e., $G_A, G_B, C_1, C_2, C_3, C_4$, and θ).

Algorithm 3 SoteriaFL-SGD, SoteriaFL-SVRG, and SoteriaFL-SAGA

Input: initial point x^0, stepsize η_t, shift stepsize γ_t, variance σ_p^2, minibatch size b, probability p

for $t = 0, 1, 2, \ldots, T$ **do**
  for each node $i \in [n]$ **do in parallel**
    Option I: SGD
      Compute local SGD estimator $\tilde{\boldsymbol{g}}_i^t = \frac{1}{b}\sum_{j\in\mathcal{I}_b} \nabla f_{i,j}(\boldsymbol{x}^t)$ // GD if choose $b = m$
    Option II: SVRG
      Compute local SVRG estimator $\tilde{\boldsymbol{g}}_i^t = \frac{1}{b}\sum_{j\in\mathcal{I}_b}(\nabla f_{i,j}(\boldsymbol{x}^t) - \nabla f_{i,j}(\boldsymbol{w}^t)) + \nabla f_i(\boldsymbol{w}^t)$
      Update SVRG snapshot point $\boldsymbol{w}^{t+1} = \begin{cases} \boldsymbol{x}^t, & \text{with probability } p \\ \boldsymbol{w}^t, & \text{with probability } 1-p \end{cases}$
    Option III: SAGA
      Compute local SAGA estimator:
$$\tilde{\boldsymbol{g}}_i^t = \frac{1}{b}\sum_{j\in\mathcal{I}_b}(\nabla f_{i,j}(\boldsymbol{x}^t) - \nabla f_{i,j}(\boldsymbol{w}_{i,j}^t)) + \frac{1}{m}\sum_{j=1}^m \nabla f_{i,j}(\boldsymbol{w}_{i,j}^t)$$
      Update SAGA variables $\boldsymbol{w}_{i,j}^{t+1} = \begin{cases} \boldsymbol{x}^t, & \text{for } j \in \mathcal{I}_b \\ \boldsymbol{w}_{i,j}^t, & \text{for } j \notin \mathcal{I}_b \end{cases}$
    End Options
    Privacy: $\boldsymbol{g}_i^t = \tilde{\boldsymbol{g}}_i^t + \boldsymbol{\xi}_i^t$, where $\boldsymbol{\xi}_i^t \sim \mathcal{N}(\boldsymbol{0}, \sigma_p^2 \boldsymbol{I})$
    Compression: let $\boldsymbol{v}_i^t = \mathcal{C}_i^t(\boldsymbol{g}_i^t - \boldsymbol{s}_i^t)$ and send to the server
    Update shift $\boldsymbol{s}_i^{t+1} = \boldsymbol{s}_i^t + \gamma_t \mathcal{C}_i^t(\boldsymbol{g}_i^t - \boldsymbol{s}_i^t)$
  end each node
  Server aggregates compressed information $\boldsymbol{v}^t = \boldsymbol{s}^t + \frac{1}{n}\sum_{i=1}^n \boldsymbol{v}_i^t$
  $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta_t \boldsymbol{v}^t$
  $\boldsymbol{s}^{t+1} = \boldsymbol{s}^t + \gamma_t \frac{1}{n}\sum_{i=1}^n \boldsymbol{v}_i^t$
end for

Lemma 1 (SGD/SVRG/SAGA estimators satisfy Assumption 3)**.** *Suppose that Assumptions 1 and 2 hold. The local SGD estimator $\tilde{\boldsymbol{g}}_i^t$ (Option I in Algorithm 3) satisfies Assumption 3 with*

$$G_A = G,\ G_B = C_1 = C_3 = C_4 = 0,\ C_2 = \frac{(m-b)G^2}{mb},\ \theta = 1,\ \Delta^t \equiv 0.$$

The local SVRG estimator $\tilde{\boldsymbol{g}}_i^t$ (Option II in Algorithm 3) satisfies Assumption 3 with

$$G_A = 2G,\ G_B = G,\ C_1 = \frac{L^2}{b},\ C_2 = 0,\ C_3 = \frac{2(1-p)\eta^2}{p},\ C_4 = 1,\ \theta = \frac{p}{2},\ \Delta^t = \|\boldsymbol{x}^t - \boldsymbol{w}^t\|^2.$$

The local SAGA estimator $\tilde{\boldsymbol{g}}_i^t$ (Option III in Algorithm 3) satisfies Assumption 3 with

$$G_A = 2G,\ G_B = G,\ C_1 = \frac{L^2}{b},\ C_2 = 0,\ C_3 = \frac{2(m-b)\eta^2}{b},\ C_4 = 1,$$
$$\theta = \frac{b}{2m},\ \Delta^t = \frac{1}{nm}\sum_{i=1}^n\sum_{j=1}^m \|\boldsymbol{x}^t - \boldsymbol{w}_{i,j}^t\|^2.$$

With Lemma 1 in hand, we can plug their corresponding parameters into the unified Theorem 3 to obtain detailed utility and communication bounds for the resulting methods (SoteriaFL-SGD/SoteriaFL-GD, SoteriaFL-SVRG, and SoteriaFL-SAGA). Formally, we have the following corollaries.

Corollary 1 (SoteriaFL-SGD/SoteriaFL-GD)**.** *Suppose that Assumption 1 and 2 hold and we combine Theorem 3 and Lemma 1, i.e., choosing stepsize $\eta_t \equiv \eta \le \frac{1}{(1+2\sqrt{(1+\omega)(1+2\omega)^2/n})L}$, where we set $\beta = \frac{(1+\omega)^2}{n(1+\tau)}$ and $\tau := \frac{(1+\omega)^{3/2}}{n^{1/2}}$, shift stepsize $\gamma_t \equiv \frac{1}{1+\omega}$, and privacy variance $\sigma_p^2 = c\frac{G^2 T\log(1/\delta)}{m^2\epsilon^2}$. If we further set the minibatch size $b = \min\left\{\frac{m\epsilon G\sqrt{\beta}}{\sqrt{(1+\omega)Ld\log(1/\delta)}}, m\right\}$ and the total number of communication rounds $T = O\Big(\frac{\sqrt{n}Lm\epsilon}{G\sqrt{(1+\omega)d\log(1/\delta)}}(1+\sqrt{\tau})\Big)$, then* SoteriaFL-SGD *satisfies (ϵ, δ)-LDP and the following utility grarantee $\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{x}_t)\|^2 \le O\Big(\frac{G\sqrt{(1+\omega)Ld\log(1/\delta)}}{\sqrt{n}m\epsilon}(1+\sqrt{\tau})\Big)$.*

Table 2: Gradient complexity for our proposed SoteriaFL-style algorithms, which is computed as the product of the total number of communication rounds T and the minibatch size b. Here, for notation simplicity, $K := \frac{\sqrt{nL}m\epsilon}{G\sqrt{(1+\omega)d\log(1/\delta)}}$ and $\tau := \frac{(1+\omega)^{3/2}}{n^{1/2}}$.

Algorithms	SoteriaFL-GD (Option I in Algorithm 3 with $b = m$)	SoteriaFL-SGD (Option I in Algorithm 3)	SoteriaFL-SVRG SoteriaFL-SAGA (Option II, III in Algorithm 3)
Gradient Complexity	$K(1+\sqrt{\tau})m$	$K(1+\sqrt{\tau})\frac{m\epsilon G\sqrt{\beta}}{\sqrt{(1+\omega)Ld\log(1/\delta)}}$	$K(1+\tau)m^{2/3}$

If we choose a minibatch size $b = m$ (local full gradient) in SoteriaFL-SGD*, the result of* SoteriaFL-SGD *leads to that of* SoteriaFL-GD.

Corollary 2 (SoteriaFL-SVRG)**.** *Suppose that Assumption 1 and 2 hold and we combine Theorem 3 and Lemma 1, i.e., choosing stepsize $\eta_t \equiv \eta \leq \frac{p^{2/3}b^{1/3}\min\{1,\sqrt{n/(1+\omega)^3}\}}{2L}$, where we set $\beta = \frac{p^{4/3}b^{2/3}(1+\omega)^2\min\{1,n/(1+\omega)^3\}}{n}$, $p^{2/3}b^{1/3} \leq 1/4$ and $p \leq 1/4$, shift stepsize $\gamma_t \equiv \frac{1}{1+\omega}$, and privacy variance $\sigma_p^2 = c\frac{5G^2T\log(1/\delta)}{m^2\epsilon^2}$. If we further let the minibatch size $b = \frac{m^{2/3}}{4}$, the probability $p = b/m$, and the total number of communication rounds $T = O\Big(\frac{\sqrt{nL}m\epsilon}{G\sqrt{(1+\omega)d\log(1/\delta)}}\max\{1,\tau\}\Big)$, where $\tau := \frac{(1+\omega)^{3/2}}{n^{1/2}}$, then* SoteriaFL-SVRG *satisfies (ϵ,δ)-LDP and the following utility guarantee $\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{x}^t)\|^2 \leq O\Big(\frac{G\sqrt{(1+\omega)Ld\log(1/\delta)}}{\sqrt{n}m\epsilon}\Big)$.*

The utility and communication complexity for SoteriaFL-SAGA are the same as SoteriaFL-SVRG, and we defer its detailed corollary to the Appendix.

Interestingly, SoteriaFL-style algorithms are more communication-efficient than CDP-SGD when the local dataset size m is large, with a communication complexity of $O(m)$, in contrast to $O(m^2)$ for CDP-SGD. In terms of utility, SoteriaFL-SVRG and SoteriaFL-SAGA can achieve the same utility as CDP-SGD, while SoteriaFL-GD and SoteriaFL-SGD achieve a slightly worse guarantee than that of CDP-SGD by a factor of $1+\sqrt{\tau}$, where $\tau := \frac{(1+\omega)^{3/2}}{n^{1/2}}$ is small when the number of clients n is large.

Gradient complexity of SoteriaFL-style algorithms Although the utility and the communication complexity are the most important considerations in private FL, another worth-noting criterion is the *gradient complexity*, which is defined as the total number of stochastic gradients computed by each client. Although SoteriaFL-GD, SoteriaFL-SGD, SoteriaFL-SVRG and SoteriaFL-SAGA have similar communication complexity (see Table 1), they actually have very different gradient complexities—summarized in Table 2—since the minibatch sizes and gradient update rules for these algorithms vary a lot. The gradient complexity of SoteriaFL-SVRG/SoteriaFL-SAGA is usually smaller than SoteriaFL-SGD, and all of them are smaller than SoteriaFL-GD. In sum, we recommend SoteriaFL-SVRG/SoteriaFL-SAGA due to its superior utility and gradient complexity while maintaining almost the same communication complexity as SoteriaFL-SGD/SoteriaFL-GD.

6 Conclusion

We propose SoteriaFL, a unified framework for private FL, which accommodates a general family of local gradient estimators including popular stochastic variance-reduced gradient methods and the state-of-the-art shifted compression scheme. A unified characterization of its performance trade-offs in terms of privacy, utility (convergence accuracy), and communication complexity is presented, which is then instantiated to arrive at several new private FL algorithms. All of these algorithms are shown to perform better than the plain CDP-SGD algorithm especially when the local dataset size is large, and have lower communication complexity compared with other private FL algorithms without compression.

References

[1] 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 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan. cpSGD: Communication-efficient and differentially-private distributed SGD. *Advances in Neural Information Processing Systems*, 31, 2018.

[3] 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*, pages 1709–1720, 2017.

[4] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 901–914, 2013.

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

[6] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[7] M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

[8] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[9] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi. Broadening the scope of differential privacy using metrics. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 82–102. Springer, 2013.

[10] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

[11] W.-N. Chen, P. Kairouz, and A. Ozgur. Breaking the communication-privacy-accuracy trilemma. *Advances in Neural Information Processing Systems*, 33:3312–3324, 2020.

[12] 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.

[13] A. Cheu, A. Smith, J. Ullman, D. Zeber, and M. Zhilyaev. Distributed differential privacy via shuffling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 375–403. Springer, 2019.

[14] A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

[15] 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, Virtual Conference*, 2021.

[16] C. Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[17] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

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

[19] V. Feldman and K. Talwar. Lossless compression of efficient private local randomizers. In *International Conference on Machine Learning*, pages 3208–3219. PMLR, 2021.

[20] V. Feldman, T. Koren, and K. Talwar. Private stochastic convex optimization: optimal rates in linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 439–449, 2020.

[21] R. C. Geyer, T. Klein, and M. Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[22] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh. Shuffled model of differential privacy in federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2521–2529. PMLR, 2021.

[23] E. Gorbunov, F. Hanzely, and P. Richtárik. Local SGD: Unified theory and new efficient methods. *arXiv preprint arXiv:2011.02828*, 2020.

[24] E. Gorbunov, K. P. Burlachenko, Z. Li, and P. Richtárik. MARINA: Faster non-convex distributed learning with compression. In *International Conference on Machine Learning*, pages 3788–3798. PMLR, 2021.

[25] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[26] S. Horváth, D. Kovalev, K. Mishchenko, S. Stich, and P. Richtárik. Stochastic distributed learning with gradient quantization and variance reduction. *arXiv preprint arXiv:1904.05115*, 2019.

[27] N. Ivkin, D. Rothchild, E. Ullah, V. Braverman, I. Stoica, and R. Arora. Communication-efficient distributed SGD with sketching. *Advances in Neural Information Processing Systems*, 32, 2019.

[28] R. Iyengar, J. P. Near, D. Song, O. Thakkar, A. Thakurta, and L. Wang. Towards practical differentially private convex optimization. In *2019 IEEE Symposium on Security and Privacy*, pages 299–316. IEEE, 2019.

[29] B. Jayaraman, L. Wang, D. Evans, and Q. Gu. Distributed learning without distress: Privacy-preserving empirical risk minimization. *Advances in Neural Information Processing Systems*, 31, 2018.

[30] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

[31] 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. *arXiv preprint arXiv:1912.04977*, 2019.

[32] P. Kairouz, Z. Liu, and T. Steinke. The distributed discrete gaussian mechanism for federated learning with secure aggregation. In *International Conference on Machine Learning*, pages 5201–5212. PMLR, 2021.

[33] S. P. Karimireddy, Q. Rebjock, S. Stich, and M. Jaggi. Error feedback fixes signSGD and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR, 2019.

[34] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[35] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson. Distributed learning with compressed gradients. *arXiv preprint arXiv:1806.06573*, 2018.

[36] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[37] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[38] D. Kovalev, S. Horváth, and P. Richtárik. Don't jump through hoops and remove those loops: Svrg and katyusha are better without the outer loop. In *Algorithmic Learning Theory*, pages 451–467. PMLR, 2020.

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[40] C.-S. Lee, N. Michelusi, and G. Scutari. Finite-bit quantization for distributed algorithms with linear convergence. *arXiv preprint arXiv:2107.11304*, 2021.

[41] Z. Li and P. Richtárik. CANITA: Faster rates for distributed convex optimization with communication compression. *Advances in Neural Information Processing Systems*, 34, 2021.

[42] Z. Li, D. Kovalev, X. Qian, and P. Richtárik. Acceleration for compressed gradient descent in distributed and federated optimization. In *International Conference on Machine Learning*, pages 5895–5904. PMLR, 2020.

[43] A. Lowy, A. Ghafelebashi, and M. Razaviyayn. Private non-convex federated learning without a trusted server. *arXiv preprint arXiv:2203.06735*, 2022.

[44] 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*, pages 1273–1282. PMLR, 2017.

[45] K. Mishchenko, E. Gorbunov, M. Takáč, and P. Richtárik. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

[46] K. Mishchenko, G. Malinovsky, S. Stich, and P. Richtárik. ProxSkip: Yes! local gradient steps provably lead to communication acceleration! finally! *arXiv preprint arXiv:2202.09357*, 2022.

[47] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani. FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization. In *International Conference on Artificial Intelligence and Statistics*, pages 2021–2031. PMLR, 2020.

[48] P. Richtárik, I. Sokolov, and I. Fatkhullin. EF21: A new, simpler, theoretically better, and practically faster error feedback. *Advances in Neural Information Processing Systems*, 34, 2021.

[49] C. Sabater, A. Bellet, and J. Ramon. Distributed differentially private averaging with improved utility and robustness to malicious parties. *arXiv preprint arXiv:2006.07218*, 2020.

[50] F. Shang, T. Xu, Y. Liu, H. Liu, L. Shen, and M. Gong. Differentially private ADMM algorithms for machine learning. *IEEE Transactions on Information Forensics and Security*, 16:4733–4745, 2021.

[51] A. T. Suresh, F. X. Yu, S. Kumar, and H. B. McMahan. Distributed mean estimation with limited communication. In *International Conference on Machine Learning*, pages 3329–3337. PMLR, 2017.

[52] A. Triastcyn, M. Reisser, and C. Louizos. DP-REC: Private & communication-efficient federated learning. *arXiv preprint arXiv:2111.05454*, 2021.

[53] T. Van Erven and P. Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[54] D. Wang, M. Ye, and J. Xu. Differentially private empirical risk minimization revisited: Faster and more general. *Advances in Neural Information Processing Systems*, 30, 2017.

[55] 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*, 31, 2018.

[56] J. Wang, Z. Charles, Z. Xu, G. Joshi, H. B. McMahan, M. Al-Shedivat, G. Andrew, S. Avestimehr, K. Daly, D. Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

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

[58] L. Wang, R. Jia, and D. Song. D2P-Fed: Differentially private federated learning with efficient communication. *arXiv preprint arXiv:2006.13039*, 2020.

[59] J. Zhang, K. Zheng, W. Mou, and L. Wang. Efficient private ERM for smooth objectives. *arXiv preprint arXiv:1703.09947*, 2017.

[60] X. Zhang, M. Fang, J. Liu, and Z. Zhu. Private and communication-efficient edge learning: a sparse differential Gaussian-masking distributed SGD approach. In *Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, pages 261–270, 2020.

[61] H. Zhao, Z. Li, and P. Richtárik. FedPAGE: A fast local stochastic gradient method for communication-efficient federated learning. *arXiv preprint arXiv:2108.04755*, 2021.

[62] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K.-Y. Lam. Local differential privacy-based federated learning for internet of things. *IEEE Internet of Things Journal*, 8(11):8836–8853, 2020.

[63] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. *Advances in Neural Information Processing Systems*, 32, 2019.

[64] H. Zong, Q. Wang, X. Liu, Y. Li, and Y. Shao. Communication reducing quantization for federated learning with local differential privacy mechanism. In *2021 IEEE/CIC International Conference on Communications in China*, pages 75–80. IEEE, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] All assumptions are stated in Section 2.
   (b) Did you include complete proofs of all theoretical results? [Yes] All detailed proofs for our theorems, lemmas and corollaries are provided in appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]