FedSKETCH: Communication-Efficient Federated Learning via Sketching

Anonymous Author(s)
Affiliation
Address
email

Abstract

Communication complexity and data privacy are the two key challenges in Federated Learning (FL) where the goal is to perform a distributed learning through a large volume of devices. In this work, we introduce two new algorithms, namely FedSKETCH and FedSKETCHGATE, to address jointly both challenges and which are, respectively, intended to be used for homogeneous and heterogeneous data distribution settings. Our algorithms are based on a key and novel sketching technique, called HEAPRIX that is unbiased, compresses the accumulation of local gradients using count sketch, and exhibits communication-efficiency properties leveraging low-dimensional sketches. We provide sharp convergence guarantees of our algorithms and validate our theoretical findings with various sets of experiments.

1 Introduction

Federated Learning (FL) is a recently emerging framework for distributed large scale machine learning problems. In FL, data is distributed across devices [23, 33] and due to privacy concerns, users are only allowed to communicate with the parameter server. Formally, the optimization problem across p distributed devices is defined as follows:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d,\ \sum_{j=1}^{p} q_j=1} f(\boldsymbol{x}) \triangleq \sum_{j=1}^{p} q_j F_j(\boldsymbol{x})\,, \tag{1}$$

where $F_j(\boldsymbol{x}) = \mathbb{E}_{\xi\in\mathcal{D}_j}\left[L_j\left(\boldsymbol{x},\xi\right)\right]$ is the local cost function at device j, $q_j \triangleq \frac{n_j}{n}$, n_j is the number of data shards at device j and $n = \sum_{j=1}^{p} n_j$ is the total number of data samples, ξ is a random variable distributed according to probability distribution $\mathcal{D}_j$, and L_j is a loss function that measures the performance of model $\boldsymbol{x}$ at device j. We note that, while for the homogeneous setting we assume $\{\mathcal{D}_j\}_{j=1}^{p}$ have the same distribution across devices and $L_i = L_j\ ,\ 1 \leq (i,j) \leq p$, in the heterogeneous setting, these distributions and loss functions L_j can vary from a device to another.

There are several challenges that need to be addressed in FL in order to efficiently learn a global model that performs well in average for all devices:

– *Communication-efficiency*: There are often many devices communicating with the server, thus incurring immense communication overhead. One approach to reduce communication round is using *local SGD with periodic averaging* [50, 39, 48, 44] which periodically averages models after a few local updates, contrary to baseline SGD [6] where gradient averaging is performed at each iteration. Local SGD has been proposed in [33, 23] under the FL setting and its convergence analysis is studied in [39, 44, 50, 48], later on improved in the followup references [3, 12, 21, 40] for homogeneous setting. It is further extended to heterogeneous setting [12, 20, 47, 30, 38, 31]. The second approach to deal with communication cost aims at reducing the size of communicated message per communication round, such as local gradient quantization [1, 4, 42, 45, 46] or sparsification [2, 32, 41, 40].

–*Data heterogeneity*: Since locally generated data in each device may come from different distribution, local computations involved in FL setting can lead to poor convergence error in practice [27, 31].

To mitigate the negative impact of data heterogeneity, [13, 16, 31, 20] suggest applying variance reduction or gradient tracking techniques along local computations.

–*Privacy* [11, 14]: Privacy has been widely addressed by injecting an additional layer of randomness to respect differential-privacy property [34] or using cryptography-based approaches under secure multi-party computation [5]. Further study of challenges can be found in recent surveys [28] and [18].

To tackle the aforementioned challenges in FL jointly, sketching based algorithms [7, 9, 22, 25] are promising approaches. For instance, to reduce communication cost, [17] develops a distributed SGD algorithm using sketching along providing its convergence analysis in the homogeneous setting, and establish a communication complexity of order $\mathcal{O}(\log(d))$ per round, where d is the dimension of the vector of parameters compared to $\mathcal{O}(d)$ complexity per round of baseline mini-batch SGD. Yet, the proposed sketching scheme in [17], built from a communication-efficiency perspective, is based on a deterministic procedure which requires access to the exact information of the gradients, thus not meeting the privacy-preserving criteria. This systemic issue is partially addressed in [37].

Focusing on privacy, [26] derives a single framework in order to tackle these issues jointly and introduces `DiffSketch` algorithm, based on the Count Sketch operator, yet does not provide its convergence analysis. Additionally, the estimation error of `DiffSketch` is higher than the sketching scheme in [17] which may end up in poor convergence.

Our main contributions are summarized as follows:

- We provide a new algorithm – `HEAPRIX` – and theoretically show that it reduces the cost of communication between devices and server, based on unbiased sketching without requiring the broadcast of exact values of gradients to the server. Based on `HEAPRIX`, we develop general algorithms for communication-efficient and sketch-based FL, namely `FedSKETCH` and `FedSKETCHGATE` for homogeneous and heterogeneous data distribution settings respectively.
- We establish non-asymptotic convergence bounds for convex, Polyak-Łojasiewicz (PL) and non-convex functions in Theorems 1 and 2 in both homogeneous and heterogeneous cases, and highlight an improvement in the number of iteration to reach a stationary point. We also provide a convergence analysis for the `PRIVIX/DiffSketch`[1] algorithm proposed in [26].
- We illustrate the benefits of `FedSKETCH` and `FedSKETCHGATE` over baseline methods through a set of experiments. The latter shows the advantages of the `HEAPRIX` compression method achieving comparable test accuracy as Federated SGD (`FedSGD`) while compressing the information exchanged between devices and server.

Notation: We denote the number of communication rounds and bits per round and per device by R and B respectively. The count sketch of vector $\boldsymbol{x}$ is designated by $\mathbf{S}(\boldsymbol{x})$. $[p]$ denotes the set $\{1, \ldots, p\}$.

2 Compression using Count Sketch

In this paper, we exploit the commonly used `Count Sketch` [7] which uses two sets of functions that encode any input vector $\boldsymbol{x}$ **into a hash table** $\boldsymbol{S}_{m\times t}(\boldsymbol{x})$. Pairwise independent hash functions $\{h_{j,1\leq j\leq t} : [d] \to m\}$ are used along with another set of pairwise independent sign hash functions $\{\text{sign}_{j,1\leq j\leq t} : [d] \to \{+1, -1\}\}$ to map entries of $\boldsymbol{x}$ ($x_i,\ 1 \leq i \leq d$) into t different columns of $\mathbf{S}_{m\times t}$, wherein to lower the dimension of the input vector we usually have $d \gg mt$. The final update reads $\mathbf{S}[j][h_j(i)] = \mathbf{S}[j][h_j(i)] + \text{sign}_j(i)x_i$ for any $1 \leq j \leq t$. There are various types of sketching algorithms which are developed based on count sketching that we develop in the following subsections. See the Appendix for the detailed Count Sketch algorithm.

2.1 Sketching based Unbiased Compressor

We define an unbiased compressor as follows:

Definition 1 (Unbiased compressor)**.** *We call randomized function, $C : \mathbb{R}^d \to \mathbb{R}^d$ an unbiased compression operator with $\Delta \geq 1$, if*

$$\mathbb{E}\left[C(\boldsymbol{x})\right] = \boldsymbol{x} \quad \textit{and} \quad \mathbb{E}\left[\|C(\boldsymbol{x})\|_2^2\right] \leq \Delta \|\boldsymbol{x}\|_2^2 \ .$$

We denote this class of compressors by $\mathbb{U}(\Delta)$.

[1] We use `PRIVIX` and `DiffSketch` [26] interchangeably throughout the paper.

This definition leads to the following property

$$\mathbb{E}\left[\|\mathrm{C}(\boldsymbol{x})-\boldsymbol{x}\|_2^2\right] \leq (\Delta-1)\|\boldsymbol{x}\|_2^2 \ .$$

Note that if we let $\Delta = 1$ then our algorithm reduces to the case of no compression. This property allows us to control the noise of the compression.

An instance of such unbiased compressor is `PRIVIX` which obtains an estimate of input $\boldsymbol{x}$ from a count sketch noted $\boldsymbol{S}(\boldsymbol{x})$. In this algorithm, to query the quantity x_i, the i-th element of the vector $\boldsymbol{x}$, we compute the median of t approximated values specified by the indices of $h_j(i)$ for $1 \leq j \leq t$, see [26], or Algorithm 6 in the Appendix (for more details). The following property of count sketch would be useful for our theoretical analysis.

Property 1 ([26])**.** *For any $\boldsymbol{x} \in \mathbb{R}^d$, we have:*

Unbiased estimation: As in [26], we have $\mathbb{E}_{\mathbf{S}}\left[\mathit{PRIVIX}[\mathbf{S}(\boldsymbol{x})]\right] = \boldsymbol{x}$.

Bounded variance: For the given $m < d$, $t = \mathcal{O}\left(\ln\left(\frac{d}{\delta}\right)\right)$ with probability $1-\delta$ we have:

$$\mathbb{E}_{\mathbf{S}}\left[\|\mathit{PRIVIX}[\mathbf{S}(\boldsymbol{x})]-\boldsymbol{x}\|_2^2\right] \leq \frac{c \times d}{m}\|\boldsymbol{x}\|_2^2 \ ,$$

where c ($e \leq c < m$) is a positive constant independent of the dimension of the input, d.

We note that bounded variance assumption does not necessary implies any compression as d could be relatively large. Thus, with probability $1-\delta$ we obtain `PRIVIX` $\in \mathbb{U}(1 + c\frac{d}{m})$. $\Delta = 1 + c\frac{d}{m}$ implies that if $m \to d$, then $\Delta \to 1 + c$, indicating a noisy reconstruction. The refrence [26] shows that if the data is normally distributed, `PRIVIX` is differentially private [10], up to additional assumptions and algorithmic design.

2.2 Sketching based Biased Compressor

A biased compressor is defined as follows:

Definition 2 (Biased compressor)**.** *A (randomized) function, $\boldsymbol{C} : \mathbb{R}^d \to \mathbb{R}^d$ belongs to $\mathbb{C}(\Delta, \alpha)$, a class of compression operators with $\alpha > 0$ and $\Delta \geq 1$, if*

$$\mathbb{E}\left[\|\alpha\boldsymbol{x}-\boldsymbol{C}(\boldsymbol{x})\|_2^2\right] \leq \left(1-\frac{1}{\Delta}\right)\|\boldsymbol{x}\|_2^2 \ ,$$

The reference [15] proves that $\mathbb{U}(\Delta) \subset \mathbb{C}(\Delta, \alpha)$. An example of biased compression via sketching and using top$_m$ operation is given below:

Algorithm 1 `HEAVYMIX`

1: **Inputs:** $\mathbf{S}(\mathbf{g})$; parameter m
2: Query the vector $\tilde{\mathbf{g}} \in \mathbb{R}^d$ from $\mathbf{S}(\mathbf{g})$:
3: Query $\hat{\ell}_2^2 = (1 \pm 0.5)\|\mathbf{g}\|^2$ from sketch $\mathbf{S}(\mathbf{g})$
4: $\forall j$ query $\hat{\mathbf{g}}_j^2 = \hat{\mathbf{g}}_j^2 \pm \frac{1}{2m}\|\mathbf{g}\|^2$ from sketch $\mathbf{S}(\mathbf{g})$
5: $H = \{j | \hat{\mathbf{g}}_j \geq \frac{\hat{\ell}_2^2}{m}\}$ and $NH = \{j | \hat{\mathbf{g}}_j < \frac{\hat{\ell}_2^2}{m}\}$
6: $\text{Top}_m = H \cup \text{rand}_\ell(NH)$, where $\ell = m - |H|$
7: Get exact values of Top_m
8: **Output:** $\tilde{\mathbf{g}} : \forall j \in \text{Top}_m : \tilde{\mathbf{g}}_i = \mathbf{g}_i$ else $\mathbf{g}_i = 0$

Following [17], `HEAVYMIX` with sketch size $\Theta\left(m \log\left(\frac{d}{\delta}\right)\right)$ is a biased compressor with $\alpha = 1$ and $\Delta = d/m$ with probability $\geq 1-\delta$, meaning that it reconstruct the $\tilde{\mathbf{g}}$ from input vector $\mathbf{g}$. In other words, with probability $1-\delta$, `HEAVYMIX` $\in C(\frac{d}{m}, 1)$. We note that Algorithm 1 is a variation of the sketching algorithm developed in [17] with distinction that `HEAVYMIX` does not require a second round of communication to obtain the exact values of top$_m$. This is mainly because in `SKETCGED-SGD` [17] the server has to obtain the exact values of *the average of sketches*; however `HEAVYMIX` obtains exact value locally, thus does not require a second round of communication. Additionally, while a sketching algorithm implementing `HEAVYMIX` has smaller estimation error compared to `PRIVIX`, it requires having access to the exact values of top$_m$, therefore not benefiting from privacy properties contrary to `PRIVIX`. In the following we introduce `HEAPRIX` which is built upon `HEAVYMIX` and `PRIVIX` methods.

2.3 Sketching based Induced Compressor

Due to Theorem 3 in [15], which illustrates that we can convert the biased compressor into an unbiased one such that, for $C_1 \in \mathbb{C}(\Delta_1)$ with $\alpha = 1$, if you choose $C_2 \in \mathbb{U}(\Delta_2)$, then induced compressor $C : x \mapsto C_1(\mathbf{x}) + C_2(\mathbf{x} - C_1(\mathbf{x}))$ belongs to $\mathbb{U}(\Delta)$ with $\Delta = \Delta_2 + \frac{1-\Delta_2}{\Delta_1}$.

Based on this notion, Algorithm 2 proposes an induced sketching algorithm by utilizing HEAVYMIX and PRIVIX for C_1 and C_2 respectively where the reconstruction of input $\mathbf{x}$ is performed using hash table $\mathbf{S}$ and $\mathbf{x}$, similar to PRIVIX and HEAVYMIX. Note that if $m \to d$, then $C(\boldsymbol{x}) \to \boldsymbol{x}$, implying that the convergence rate can be improved by decreasing the size of compression m.

Algorithm 2 HEAPRIX

1: **Inputs:** $\boldsymbol{x} \in \mathbb{R}^d, t, m, \mathbf{S}_{m\times t}, h_j(1 \leq i \leq t), \text{sign}_j(1 \leq i \leq t)$, parameter m

2: Approximate $\mathbf{S}(\boldsymbol{x})$ using HEAVYMIX

3: Approximate $\mathbf{S}\left(\boldsymbol{x} - \texttt{HEAVYMIX}[\mathbf{S}(\boldsymbol{x})]\right)$ with PRIVIX

4: **Output:**

$$\texttt{HEAVYMIX}\left[\mathbf{S}\left(\boldsymbol{x}\right)\right] + \texttt{PRIVIX}\left[\mathbf{S}\left(\boldsymbol{x} - \texttt{HEAVYMIX}\left[\mathbf{S}\left(\boldsymbol{x}\right)\right]\right)\right].$$

Corollary 1. *Based on Theorem 3 of [15], HEAPRIX in Algorithm 2 satisfies* $C(\boldsymbol{x}) \in \mathbb{U}(c\frac{d}{m})$.

Benefits of HEAPRIX: Corollary 1 states that, unlike PRIVIX, HEAPRIX compression noise can be made as small as possible using larger hash size. In the distributed setting, contrary to SKETCHED-SGD [17] where decompressing is happening at the server, HEAPRIX does not require having access to exact top_m values of the input as it is based on HEAVYMIX, which helps preserving privacy. In other words, HEAPRIX leverages the best of both: the *unbiasedness* of PRIVIX while using *heavy hitters* as in HEAVYMIX.

3 FedSKETCH and FedSKETCHGATE

We introduce two new algorithms for both homogeneous and heterogeneous settings.

3.1 Homogeneous Setting

In FedSKETCH, the number of local updates, between two consecutive communication rounds, at device j is denoted by τ. Unlike [13], server node does not store any global model, rather, device j has two models: $\boldsymbol{x}^{(r)}$ and $\boldsymbol{x}_j^{(\ell,r)}$, which are respectively the local and global models. We develop FedSKETCH in Algorithm 3. A variant of this algorithm implementing HEAPRIX is also described in Algorithm 3. We remark that for this variant, we need to have an additional communication round between server and worker j to aggregate $\delta_j^{(r)} \triangleq \mathbf{S}_j\left[\texttt{HEAVYMIX}(\mathbf{S}^{(r)})\right]$ (Lines 3 and 3) to compute $\mathbf{S}^{(r)} = \frac{1}{k}\sum_{j\in\mathcal{K}}\mathbf{S}_j^{(r)}$. The main difference between FedSKETCH and DiffSketch in [26] is that we use distinct local and global learning rates. Furthermore, unlike [26], we do not add local Gaussian noise.

Algorithm 3 FedSKETCH(R, τ, η, γ)

1: **Inputs:** $\boldsymbol{x}^{(0)}$: initial model shared by local devices, global and local learning rates γ and η, respectively

2: **for** $r = 0, \ldots, R-1$ **do**

3: **parallel for device** $j \in \mathcal{K}^{(r)}$ **do**:

4: **if PRIVIX variant:**

$$\boldsymbol{\Phi}^{(r)} \triangleq \texttt{PRIVIX}\left[\mathbf{S}^{(r-1)}\right]$$

5: **if HEAPRIX variant:**

$$\boldsymbol{\Phi}^{(r)} \triangleq \texttt{HEAVYMIX}\left[\mathbf{S}^{(r-1)}\right] + \texttt{PRIVIX}\left[\mathbf{S}^{(r-1)} - \tilde{\mathbf{S}}^{(r-1)}\right]$$

6: Set $\boldsymbol{x}^{(r)} = \boldsymbol{x}^{(r-1)} - \gamma\boldsymbol{\Phi}^{(r)}$ and $\boldsymbol{x}_j^{(0,r)} = \boldsymbol{x}^{(r)}$

7: **for** $\ell = 0, \ldots, \tau - 1$ **do**

8: Sample a mini-batch $\xi_j^{(\ell,r)}$ and compute $\tilde{\mathbf{g}}_j^{(\ell,r)}$

9: Update $\boldsymbol{x}_j^{(\ell+1,r)} = \boldsymbol{x}_j^{(\ell,r)} - \eta\,\tilde{\mathbf{g}}_j^{(\ell,r)}$

10: **end for**

11: Device j broadcasts $\mathbf{S}_j^{(r)} \triangleq \mathbf{S}_j\left(\boldsymbol{x}_j^{(0,r)} - \boldsymbol{x}_j^{(\tau,r)}\right)$.

12: Server **computes** $\mathbf{S}^{(r)} = \frac{1}{k}\sum_{j\in\mathcal{K}}\mathbf{S}_j^{(r)}$.

13: Server **broadcasts** $\mathbf{S}^{(r)}$ to devices in randomly drawn devices $\mathcal{K}^{(r)}$.

14: **if HEAPRIX variant:**

15: Second round of communication: $\delta_j^{(r)} := \mathbf{S}_j\left[\texttt{HEAVYMIX}(\mathbf{S}^{(r)})\right]$ and broadcasts $\tilde{\mathbf{S}}^{(r)} \triangleq \frac{1}{k}\sum_{j\in\mathcal{K}}\delta_j^{(r)}$ to devices in set $\mathcal{K}^{(r)}$

16: **end parallel for**

17: **end**

18: **Output:** $\boldsymbol{x}^{(R-1)}$

Algorithmic comparison with [13] An important feature of our algorithm is that due to a lower dimension of the count sketch, the resulting averages ($\mathbf{S}^{(r)}$ and $\tilde{\mathbf{S}}^{(r)}$) received by the server are also of lower dimension. Therefore, these algorithms exploit a bidirectional compression during the communication from server to device back and forth. As a result, for the case of large quantization error $\omega = \theta(\frac{d}{m})$ as shown in [13], our algorithms can outperform FedCOM and FedCOMGATE developed in [13] if sufficiently large hash tables are used and the uplink communication cost is high. Furthermore, while, in [13], server stores a global model and aggregates the partial gradients from devices which can enable the server to extract some information regarding the device's data, in contrast, in our algorithms server does not store the global model and only broadcasts the average sketches. Thus, sketching-based server-devices communication algorithms such as ours do not reveal the exact values of the inputs, to preserve privacy as a by-product.

Remark 1. *As pointed out in [15], while induced compressors transform a biased compressor into unbiased one, as a drawback it doubles communication cost since the devices need to send $C_1(\boldsymbol{x})$ and $C_2\left(\boldsymbol{x} - C_1\left(\boldsymbol{x}\right)\right)$ separately. We note that in the special case of* `HEAPRIX`*, due to the use of sketching, the extra communication round cost is compensated with lower number of bits per round thanks to the lower dimension of sketching.*

3.2 Heterogeneous Setting

In this section, we focus on the optimization problem of (1) in the special case of $q_1 = \ldots = q_p = \frac{1}{p}$ with full device participation ($k = p$). These results can be extended to the scenario where devices are sampled. For non i.i.d. data, the `FedSKETCH` algorithm, designed for homogeneous setting, may fail to perform well in practice. The main reason is that in FL, devices are using local stochastic descent direction which could be different than global descent direction when the data distribution are non-identical. Therefore, to mitigate the effect of data heterogeneity, we introduce a new algorithm called `FedSKETCHGATE` described in Algorithm 4. This algorithm leverages the idea of gradient tracking applied in [13] (with compression) and a special case of $\gamma = 1$ without compression [31]. The main idea is that using an approximation of global gradient, $\mathbf{c}_j^{(r)}$ allows to correct the local gradient direction. For the `FedSKETCHGATE` with `PRIVIX` variant, the correction vector $\mathbf{c}_j^{(r)}$ at device j and communication round r is computed in Line 4. While using `HEAPRIX` compression, `FedSKETCHGATE` also updates $\tilde{\mathbf{S}}^{(r)}$ via Line 4.

Algorithm 4 `FedSKETCHGATE`(R, τ, η, γ)

1: **Inputs:** $\boldsymbol{x}^{(0)} = \boldsymbol{x}_j^{(0)}$ shared by all local devices, global and local learning rates γ and η.
2: **for** $r = 0, \ldots, R-1$ **do**
3: **parallel for device** $j = 1, \ldots, p$ **do**:
4: **if PRIVIX variant:**

$$\mathbf{c}_j^{(r)} = \mathbf{c}_j^{(r-1)} - \frac{1}{\tau}\left[\mathtt{PRIVIX}\left(\mathbf{S}^{(r-1)}\right) - \mathtt{PRIVIX}\left(\mathbf{S}_j^{(r-1)}\right)\right]$$

where $\mathbf{\Phi}^{(r)} \triangleq \mathtt{PRIVIX}(\mathbf{S}^{(r-1)})$
5: **if HEAPRIX variant:**

$$\mathbf{c}_j^{(r)} = \mathbf{c}_j^{(r-1)} - \frac{1}{\tau}\left(\mathbf{\Phi}^{(r)} - \mathbf{\Phi}_j^{(r)}\right)$$

6: Set $\boldsymbol{x}^{(r)} = \boldsymbol{x}^{(r-1)} - \gamma \mathbf{\Phi}^{(r)}$ and $\boldsymbol{x}_j^{(0,r)} = \boldsymbol{x}^{(r)}$
7: **for** $\ell = 0, \ldots, \tau - 1$ **do**
8: Sample mini-batch $\xi_j^{(\ell,r)}$ and compute $\tilde{\mathbf{g}}_j^{(\ell,r)}$
9: $\boldsymbol{x}_j^{(\ell+1,r)} = \boldsymbol{x}_j^{(\ell,r)} - \eta\left(\tilde{\mathbf{g}}_j^{(\ell,r)} - \mathbf{c}_j^{(r)}\right)$
10: **end for**
11: Device j broadcasts $\mathbf{S}_j^{(r)} \triangleq \mathbf{S}\left(\boldsymbol{x}_j^{(0,r)} - \boldsymbol{x}_j^{(\tau,r)}\right)$.
12: Server **computes** $\mathbf{S}^{(r)} = \frac{1}{p}\sum_{j=1} \mathbf{S}_j^{(r)}$ and **broadcasts** $\mathbf{S}^{(r)}$ to all devices.
13: **if HEAPRIX variant:**
14: Device j computes $\mathbf{\Phi}_j^{(r)} \triangleq \mathtt{HEAPRIX}[\mathbf{S}_j^{(r)}]$
15: Second round of communication to obtain $\delta_j^{(r)} := \mathbf{S}_j\left(\mathtt{HEAVYMIX}[\mathbf{S}^{(r)}]\right)$
16: Broadcasts $\tilde{\mathbf{S}}^{(r)} \triangleq \frac{1}{p}\sum_{j=1}^{p} \delta_j^{(r)}$ to devices
17: **end parallel for**
18: **end**
19: **Output:** $\boldsymbol{x}^{(R-1)}$

Remark 2. *Most of the existing communication-efficient algorithms with compression only consider communication-efficiency from devices to server. However, Algorithms 3 and 4 also improve the communication efficiency from server to devices since it exploits low-dimensional sketches (and averages), communicated from the server to devices.*

For both `FedSKETCH` and `FedSKETCHGATE` algorithms, unlike `PRIVIX`, `HEAPRIX` variant requires a second round of communication. Therefore, in Cross-Device FL setting, where there could be millions of devices, `HEAPRIX` variant may not be practical, and we note that it could be more suitable for Cross-Silo FL setting.

4 Convergence Analysis

We first state commonly used assumptions required in the following convergence analysis (reminder of our notations can be found Table 1 of the Appendix).

Assumption 1 (Smoothness and Lower Boundedness)**.** *The local objective function $f_j(\cdot)$ of device j is differentiable for $j \in [p]$ and L-smooth, i.e., $\|\nabla f_j(\boldsymbol{x}) - \nabla f_j(\mathbf{y})\| \leq L\|\boldsymbol{x} - \mathbf{y}\|$, $\forall\, \boldsymbol{x}, \mathbf{y} \in \mathbb{R}^d$. Moreover, the optimal objective function $f(\cdot)$ is bounded below by $f^* := \min_{\boldsymbol{x}} f(\boldsymbol{x}) > -\infty$.*

Assumption 1 is common in stochastic optimization. We present our results for PL, convex and general non-convex objectives. [19] show that PL condition implies strong convexity property with same module (PL objectives can also be non-convex, hence strong convexity does not imply PL condition necessarily).

4.1 Convergence of `FEDSKETCH`

We now focus on the homogeneous case where data is i.i.d. among local devices, and therefore , the stochastic local gradient of each worker is an unbiased estimator of the global gradient. We have:

Assumption 2 (Bounded Variance)**.** *For all $j \in [m]$, we can sample an independent mini-batch ℓ_j of size $|\Xi_j^{(\ell,r)}| = b$ and compute an unbiased stochastic gradient $\tilde{\mathbf{g}}_j = \nabla f_j(\boldsymbol{x}; \Xi_j)$, $\mathbb{E}_{\xi_j}[\tilde{\mathbf{g}}_j] = \nabla f(\boldsymbol{x}) = \mathbf{g}$ with the variance bounded is bounded by a constant σ^2, i.e., $\mathbb{E}_{\Xi_j}\left[\|\tilde{\mathbf{g}}_j - \mathbf{g}\|^2\right] \le \sigma^2$.*

Theorem 1. *Suppose Assumptions 1-2 hold. Given $0 < m \le d$ and considering Algorithm 3 with sketch size $B = O\left(m \log\left(\frac{dR}{\delta}\right)\right)$ and $\gamma \ge k$, with probability $1-\delta$ we have:*

*In the **non-convex** case, $\{\boldsymbol{x}^{(r)}\}_{r=>0}$ satisfies $\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}\left[\left\|\nabla f(\boldsymbol{x}^{(r)})\right\|_2^2\right] \le \epsilon$ if:*

- *`FS-PRIVIX`, for $\eta = \frac{1}{L\gamma}\sqrt{\frac{k}{R\tau\left(\frac{cd}{mk}+1\right)}}$: $R = O\left(1/\epsilon\right)$ and $\tau = O\left((d+m)/(mk\epsilon)\right)$.*
- *`FS-HEAPRIX`, for $\eta = \frac{1}{L\gamma}\sqrt{\frac{k}{R\tau\left(\frac{cd-m}{mk}+1\right)}}$: $R = O\left(1/\epsilon\right)$ and $\tau = O\left(d/(mk\epsilon)\right)$.*

*In the **PL or strongly convex** case, $\{\boldsymbol{x}^{(r)}\}_{r=>0}$ satisfies $\mathbb{E}[f(\boldsymbol{x}^{(R-1)}) - f(\boldsymbol{x}^{(*)})] \le \epsilon$ if we set:*

- *`FS-PRIVIX`, for $\eta = \frac{1}{2L(cd/mk+1)\tau\gamma}$: $R = O\left(\left(d/mk+1\right)\kappa\log\left(1/\epsilon\right)\right)$ and $\tau = O\left(\left(d/m+1\right)\Big/\left(d/m+k\right)\epsilon\right)$.*
- *`FS-HEAPRIX`, for $\eta = \frac{1}{2L((cd-m)/mk+1)\tau\gamma}$: $R = O\left(\left((d-m)/mk+1\right)\kappa\log\left(1/\epsilon\right)\right)$ and $\tau = O\left(d/m\Big/\left(\left((d/m-1)+k\right)\epsilon\right)\right)$.*

*In the **Convex** case, $\{\boldsymbol{x}^{(r)}\}_{r=>0}$ satisfies $\mathbb{E}\Big[f(\boldsymbol{x}^{(R-1)}) - f(\boldsymbol{x}^{(*)})\Big] \le \epsilon$ if we set:*

- *`FS-PRIVIX`, for $\eta = \frac{1}{2L(cd/mk+1)\tau\gamma}$: $R = O\left(L\left(1+d/mk\right)/\epsilon\log\left(1/\epsilon\right)\right)$ and $\tau = O\left(\left(d/m+1\right)^2/\left(k\left(d/mk+1\right)^2\epsilon^2\right)\right)$.*
- *`FS-HEAPRIX`, for $\eta = \frac{1}{2L((cd-m)/mk+1)\tau\gamma}$: $R = O\left(L\left(1+(d-m)/mk\right)/\epsilon\log\left(1/\epsilon\right)\right)$ and $\tau = O\left((d/m)^2/\left(k\left([d-m]/mk+1\right)^2\epsilon^2\right)\right)$.*

The bounds in Theorem 1 suggest that in homogeneous setting if we set $d = m$ (no compression), the number of communication rounds to achieve the ϵ error matches with the number of iterations required to achieve the same error under a centralized setting. Additionally, computational complexity scales down with number of sampled devices. To stress on the further impact of using sketching, we also compare our results with prior works in terms of total number of communicated bits per device.

Comparison with [17] From privacy aspect, we note [17] requires for server to have access to exact values of top_m gradients, hence do not preserve privacy, whereas our schemes do not need those exact values. From communication cost point of view, for strongly convex objective and compared to [17], we improve the total communication per worker from $RB = O\left(\frac{d}{\epsilon}\log\left(\frac{d}{\delta\sqrt{\epsilon}}\max\left(\frac{d}{m}, \frac{1}{\sqrt{\epsilon}}\right)\right)\right)$ to

$$RB = O\left(\kappa(\tfrac{d-m}{k} + m)\log\tfrac{1}{\epsilon}\log\left(\tfrac{\kappa d}{\delta}(\tfrac{d-m}{mk}+1)\log\tfrac{1}{\epsilon}\right)\right).$$

We note that while reducing communication cost, our scheme requires $\tau = O(d/m(k(\frac{d}{mk}+1)\epsilon)) > 1$, which scales down with the number of sampled devices, k. Moreover, unlike [17], we do not use bounded gradient assumption. Therefore, we obtain stronger result with weaker assumptions. Regarding general non-convex objectives, our result improves the total communication cost per worker in [17] from $RB = O\left(\max(\frac{1}{\epsilon^2}, \frac{d^2}{k^2\epsilon}\right)\log(\frac{d}{\delta}\max(\frac{1}{\epsilon^2}, \frac{d^2}{k^2\epsilon})))$ for *only one device* to $RB =$

$O(\frac{m}{\epsilon}\log(\frac{d}{\epsilon\delta}))$. We also highlight that we can obtain similar rates for Algorithm 3 in heterogeneous environment if we make the additional assumption of uniformly bounded gradient.

Note: Such improved communication cost over prior related works is due to joint exploitation of *sketching*, to reduce the dimension of communicated messages, and the use of *local updates*, to reduce the total number of communication rounds leading to a specific convergence error.

4.2 Convergence of `FedSKETCHGATE`

We start with bounded local variance assumption:

Assumption 3 (Bounded Local Variance). *For all $j \in [p]$, we can sample an independent mini-batch Ξ_j of size $|\xi_j| = b$ and compute an unbiased stochastic gradient $\tilde{\mathbf{g}}_j = \nabla f_j(\boldsymbol{x};\Xi_j)$ with $\mathbb{E}_\xi[\tilde{\mathbf{g}}_j] = \nabla f_j(\boldsymbol{x}) = \mathbf{g}_j$. Moreover, the variance of local stochastic gradients is bounded such that $\mathbb{E}_\Xi\left[\|\tilde{\mathbf{g}}_j - \mathbf{g}_j\|^2\right] \le \sigma^2$.*

Theorem 2. *Suppose Assumptions 1 and 3 hold. Given $0 < m \le d$, and considering `FedSKETCHGATE` in Algorithm 4 with sketch size $B = O\left(m\log\left(\frac{dR}{\delta}\right)\right)$ and $\gamma \ge p$ with probability $1-\delta$ we have*

*In the **non-convex** case, $\eta = \frac{1}{L\gamma}\sqrt{\frac{mp}{R\tau(cd)}}$, $\{\boldsymbol{x}^{(r)}\}_{r=>0}$ satisfies $\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}\left[\left\|\nabla f(\boldsymbol{x}^{(r)})\right\|_2^2\right] \le \epsilon$ if:*

- *`FS-PRIVIX`:*

$$R = O((d+m)/m\epsilon) \quad \textit{and} \quad \tau = O(1/(p\epsilon))\ .$$

- *`FS-HEAPRIX`: $R = O(d/m\epsilon)$ and $\tau = O(1/(p\epsilon))$.*

*In the **PL or Strongly convex** case, $\{\boldsymbol{x}^{(r)}\}_{r=>0}$ satisfies $\mathbb{E}\Big[f(\boldsymbol{x}^{(R-1)}) - f(\boldsymbol{x}^{(*)})\Big] \le \epsilon$ if:*

- *`FS-PRIVIX`, for $\eta = 1/(2L(\frac{cd}{m}+1)\tau\gamma)$: $R = O\left((\frac{d}{m}+1)\kappa\log(1/\epsilon)\right)$ and $\tau = O\left(1/(p\epsilon)\right)$*
- *`FS-HEAPRIX`, for $\eta = m/(2cLd\tau\gamma)$: $R = O\left((\frac{d}{m})\kappa\log(1/\epsilon)\right)$ and $\tau = O\left(1/(p\epsilon)\right)$.*

*In the **convex** case, $\{\boldsymbol{x}^{(r)}\}_{r=>0}$ satisfies $\mathbb{E}[f(\boldsymbol{x}^{(R-1)}) - f(\boldsymbol{x}^{(*)})] \le \epsilon$ if:*

- *`FS-PRIVIX`, for $\eta = 1/(2L(cd/m+1)\tau\gamma)$: $R = O\left(L(d/m+1)\epsilon\log(1/\epsilon)\right)$ and $\tau = O\left(1/(p\epsilon^2)\right)$.*
- *`FS-HEAPRIX`, for $\eta = m/(2Lcd\tau\gamma)$: $R = O\left(L(d/m)\epsilon\log(1/\epsilon)\right)$ and $\tau = O\left(1/(p\epsilon^2)\right)$.*

Theorem 2 implies that the number of communication rounds and local updates are similar to the corresponding quantities in homogeneous setting except for the non-convex case where the number of rounds also depends on the compression rate (summarized Table 2-3 of the Appendix).

4.3 Comparison with Prior Methods

Before comparing with prior works, we highlight that privacy is another purpose of using unbiased sketching in addition to communication efficiency. Therefore, our main competing schemes are distributed algorithms based on sketching. Nonetheless, for the sake of showing the effectiveness of our algorithms, we also compare with prior non-sketching based distributed algorithms ([20, 3, 36, 13]) in Section B of Appendix.

Comparison with [26]. Note that our convergence analysis does not rely on the bounded gradient assumption. We also improve both the number of communication rounds R and the size of transmitted bits B per communication round. Additionally, we highlight that, while [26] provides a convergence analysis for convex objectives, our analysis holds for PL (thus strongly convex case), general convex and general non-convex objectives.

Comparison with [37]. Due to gradient tracking, our algorithm tackles data heterogeneity issue, while algorithms in [37] does not particularly. As a consequence, in `FedSKETCHGATE` each device has to store an additional state vector compared to [37]. Yet, as our method is built upon an unbiased compressor, server does not need to store any additional error correction vector. The convergence results for both of two variants of `FetchSGD` in [37] rely on the uniform bounded gradient assumption which may not be applicable with L-smoothness assumption when data distribution is highly heterogeneous, as in FL, see [21], while our bounds do not assume such boundedness.

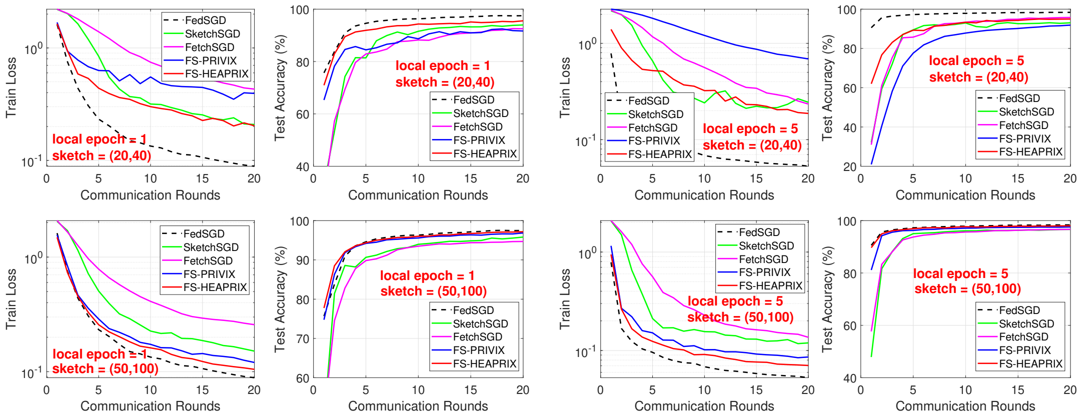

Figure 1: Homogeneous case: Comparison of compressed optimization methods on LeNet CNN.

Besides, Theorem 1 [37] assumes that *Contraction Holds* for the sequence of gradients which may not hold in practice, yet based on this strong assumption, their total communication cost (RB) in order to achieve ϵ error is $RB = O\left(m\max(\frac{1}{\epsilon^2},\frac{d^2-dm}{m^2\epsilon})\log\left(\frac{d}{\delta}\max(\frac{1}{\epsilon^2},\frac{d^2-dm}{m^2\epsilon})\right)\right)$. For the sake of comparison we let the compression ratio in [37] to be $\frac{m}{d}$. In contrast, without any extra assumptions, our results in Theorem 2 for `PRIVIX` and `HEAPRIX` are respectively $RB = O(\frac{(d+m)}{\epsilon}\log(\frac{(\frac{d^2}{m})+d}{\epsilon\delta}))$ and $RB = O(\frac{d}{\epsilon}\log(\frac{d^2}{\epsilon m\delta}))$ which improves the total communication cost of Theorem 1 in [37] under regimes such that $\frac{1}{\epsilon} \geq d$ or $d \gg m$. Theorem 2 in [37] is based the *Sliding Window Heavy Hitters* assumption, which is similar to the gradient diversity assumption in [29, 12]. Under that assumption the total communication cost is shown to be $RB = O\left(\frac{m\max(I^{2/3},2-\alpha)}{\epsilon^3\alpha}\log\left(\frac{d\max(I^{2/3},2-\alpha)}{\epsilon^3\delta}\right)\right)$ where I is a constant related to the window of gradients. We improve this bound under weaker assumptions in a regime where $\frac{I^{2/3}}{\epsilon^2} \geq d$. We also provide bounds for PL, convex and non-convex objectives contrary to [37]. Finally, we note that algorithms in [37] are using momentum at server. While we do not use it explicitly, we can modify our algorithms to include momentum easily.

5 Numerical Study

In this section, we provide empirical results on MNIST benchmark dataset to demonstrate the effectiveness of our proposed algorithms. We train LeNet-5 Convolutional Neural Network (CNN) architecture introduced in [24], with 60 000 parameters. We compare Federated SGD (`FedSGD`) as the full-precision baseline, along with four sketching methods `SketchSGD` [17], `FetchSGD` [37], and two FedSketch variants `FS-PRIVIX` and `FS-HEAPRIX`. Note that in Algorithm 3, `FS-PRIVIX` with global learning rate $\gamma = 1$ is equivalent to the `DiffSketch` algorithm proposed in [29]. Also, `SketchSGD` is slightly modified to compress the change in local weights (instead of local gradient in every iteration), and `FetchSGD` is implemented with second round of communication for fairness. (The original proposal does not include second round of communication, which performs worse with small sketch size.) As suggested in [37], the momentum factor of `FetchSGD` is set to 0.9, and we also follow some recommended implementation tricks to improve its performance, which are detailed in the Appendix. The number of workers is set to 50 and we report the results for 1 and 5 local epochs. A local epoch is finished when all workers go through their local data samples once. The local batch size is 30. In each round, we randomly choose half of the devices to be active. We tune the learning rates (η and γ, if applicable) over log-scale and report the best results, for both *homogeneous* and *heterogeneous* setting. In the former case, each device receives uniformly drawn data samples, and in the latter, it only receives samples from one or two classes among ten.

Homogeneous case. In Figure 1, we provide the training loss and test accuracy with different number of local epochs and sketch size, $(t, k) = (20, 40)$ and $(50, 100)$. Note that, these two choices of sketch size correspond to a $75\times$ and $12\times$ compression ratio, respectively. We conclude

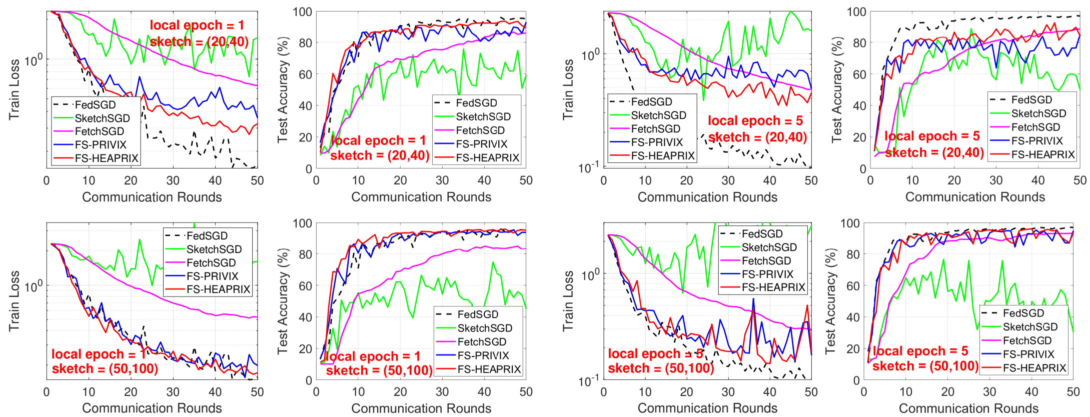

Figure 2: Heterogeneous case: Comparison of compressed optimization algorithms on LeNet CNN.

- In general, increasing compression ratio would sacrifice learning performance. In all cases, `FS-HEAPRIX` performs the best in terms of both training objective and test accuracy, among all compressed methods.
- `FS-HEAPRIX` is better than `FS-PRIVIX`, especially with small sketches (high compression ratio). `FS-HEAPRIX` yields acceptable extra test error compared to full-precision `FedSGD`, particularly when considering the high compression ratio (e.g., $75\times$).
- From the training loss, we see that the performance of `FS-HEAPRIX` improves when the number of local updates increases. *That is, the proposed method is able to further reduce the communication cost by reducing the number of rounds required for communication.* This is also consistent with our theoretical findings.

In general, our proposed `FS-HEAPRIX` outperforms all competing methods, and a sketch size of $(50, 100)$ is sufficient to approach the accuracy of full-precision `FedSGD`.

Heterogeneous case. We plot similar set of results in Figure 2 for non-i.i.d. data distribution, which leads to more twists and turns in the training curves. We see that `SketchSGD` performs very poorly in the heterogeneous case, which is improved by error tracking and momentum in `FetchSGD`, as expected. However, both of these methods are worse than our proposed `FedSketchGATE` methods, which can achieve similar generalization accuracy as full-precision `FedSGD`, even with small sketch size (i.e., $75\times$ compression with 1 local epoch). Note that, slower convergence and worse generalization of `FedSGD` in non-i.i.d. data distribution case is also reported in e.g. [33, 8].

We also notice in Figure 2 the edge of `FS-HEAPRIX` over `FS-PRIVIX` in terms of training loss and test accuracy. However, we see that in the heterogeneous setting, more local updates tend to undermine the learning performance, especially with small sketch size. Nevertheless, when the sketch size is not too small, i.e., $(50, 100)$, `FS-HEAPRIX` can still provide comparable test accuracy as `FedSGD` in both cases. Our empirical study demonstrates that `FedSketch` (and `FedSketchGATE`) frameworks are able to perform well in homogeneous (resp. heterogeneous) settings, with high compression rate. In particular, `FedSketch` methods are beneficial over `SketchedSGD` [17] and `FetchSGD` [37] in all cases. `FS-HEAPRIX` performs the best among all the tested compressed algorithms, which in many cases achieves similar generalization accuracy as full-precision FedSGD with small sketch size.

6 Conclusion

In this paper, we introduced `FedSKETCH` and `FedSKETCHGATE` algorithms for homogeneous and heterogeneous data distribution setting respectively for Federated Learning wherein communication between server and devices is only performed using count sketch. Our algorithms, thus, provide communication-efficiency and privacy, through random hashes based sketches. We analyze the convergence error for *non-convex*, *PL* and *general convex* objective functions in the scope of Federated Optimization. We provide insightful numerical experiments showcasing the advantages of our `FedSKETCH` and `FedSKETCHGATE` methods over current federated optimization algorithm. The proposed algorithms outperform competing compression method and can achieve comparable test accuracy as Federated SGD, with high compression ratio.

References

[1] 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 (NIPS)*, pages 1709–1720, Long Beach, 2017.

[2] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5973–5983, Montréal, Canada, 2018.

[3] D. Basu, D. Data, C. Karakus, and S. N. Diggavi. Qsparse-local-sgd: Distributed SGD with quantization, sparsification and local computations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14668–14679, Vancouver, Canada, 2019.

[4] J. Bernstein, Y. Wang, K. Azizzadenesheli, and A. Anandkumar. SIGNSGD: compressed optimisation for non-convex problems. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 559–568, Stockholmsmässan, Stockholm, Sweden, 2018.

[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1175–1191, Dallas, TX, 2017.

[6] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 161–168, Vancouver, Canada, 2008.

[7] M. Charikar, K. C. Chen, and M. Farach-Colton. Finding frequent items in data streams. *Theoretical Computer Science*, 312(1):3–15, 2004. doi: 10.1016/S0304-3975(03)00400-6. URL `https://doi.org/10.1016/S0304-3975(03)00400-6`.

[8] X. Chen, X. Li, and P. Li. Toward communication efficient adaptive gradient method. In *ACM-IMS Foundations of Data Science Conference (FODS)*, Seattle, WA, 2020.

[9] G. Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005.

[10] C. Dwork. Differential privacy. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.

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

[12] F. Haddadpour and M. Mahdavi. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019.

[13] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi. Federated learning with compression: Unified analysis and sharp guarantees. *arXiv preprint arXiv:2007.01154*, 2020.

[14] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*, 2017.

[15] S. Horváth and P. Richtárik. A better alternative to error feedback for communication-efficient distributed learning. *arXiv preprint arXiv:2006.11077*, 2020.

[16] 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.

[17] N. Ivkin, D. Rothchild, E. Ullah, V. Braverman, I. Stoica, and R. Arora. Communication-efficient distributed SGD with sketching. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 13144–13154, Vancouver, Canada, 2019.

[18] 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.

[19] H. Karimi, J. Nutini, and M. Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pages 795–811, Riva del Garda, Italy, 2016.

[20] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh. Scaffold: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

[21] A. Khaled, K. Mishchenko, and P. Richtárik. Tighter theory for local SGD on identical and heterogeneous data. In *The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 4519–4529, Online [Palermo, Sicily, Italy], 2020.

[22] J. Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

[23] 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.

[24] 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.

[25] P. Li, K. W. Church, and T. Hastie. One sketch for all: Theory and application of conditional random sampling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 953–960, Vancouver, Canada, 2008.

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

[27] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.*, 37(3):50–60, 2020.

[28] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[29] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems (MLSys)*, Austin, TX, 2020.

[30] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. On the convergence of fedavg on non-iid data. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, 2020.

[31] X. Liang, S. Shen, J. Liu, Z. Pan, E. Chen, and Y. Cheng. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019.

[32] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[33] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282, Fort Lauderdale, FL, 2017.

[34] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[35] C. Philippenko and A. Dieuleveut. Artemis: tight convergence guarantees for bidirectional compression in federated learning. *arXiv preprint arXiv:2006.14591*, 2020.

[36] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2021–2031, Online [Palermo, Sicily, Italy], 2020.

[37] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora. FetchSGD: Communication-efficient federated learning with sketching. *arXiv preprint arXiv:2007.07682*, 2020.

[38] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith. On the convergence of federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[39] S. U. Stich. Local sgd converges fast and communicates little. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, LA, 2019.

[40] S. U. Stich and S. P. Karimireddy. The error-feedback framework: Better rates for sgd with delayed gradients and compressed communication. *arXiv preprint arXiv:1909.05350*, 2019.

[41] S. U. Stich, J.-B. Cordonnier, and M. Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4447–4458, Montréal, Canada, 2018.

[42] H. Tang, S. Gan, C. Zhang, T. Zhang, and J. Liu. Communication compression for decentralized training. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 7652–7662, Montréal, Canada, 2018.

[43] H. Tang, C. Yu, X. Lian, T. Zhang, and J. Liu. Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. In *International Conference on Machine Learning*, pages 6155–6165. PMLR, 2019.

[44] J. Wang and G. Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

[45] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in neural information processing systems (NIPS)*, pages 1509–1519, Long Beach, CA, 2017.

[46] J. Wu, W. Huang, J. Huang, and T. Zhang. Error compensated quantized sgd and its applications to large-scale distributed optimization. *arXiv preprint arXiv:1806.08054*, 2018.

[47] H. Yu, R. Jin, and S. Yang. On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 7184–7193, Long Beach, CA, 2019.

[48] H. Yu, S. Yang, and S. Zhu. Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, pages 5693–5700, Honolulu, HI, 2019.

[49] S. Zheng, Z. Huang, and J. T. Kwok. Communication-efficient distributed blockwise momentum sgd with error-feedback. *arXiv preprint arXiv:1905.10936*, 2019.

[50] F. Zhou and G. Cong. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3219–3227, Stockholm, Sweden, 2018.

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? answerYes
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (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]
   (b) Did you include complete proofs of all theoretical results? [Yes]
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)? [No] Available upon demand.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Runs are averaged over several runs.
   (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? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (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]