Federated Learning with Compression: Unified Analysis and Sharp Guarantees

Farzin Haddadpour
Penn State University

Mohammad Mahdi Kamani
Penn State University

Aryan Mokhtari
Texas University at Austin

Mehrdad Mahdavi
Penn State University

Abstract

In federated learning, communication cost is often a critical bottleneck to scale up distributed optimization algorithms to collaboratively learn a model from millions of devices with potentially unreliable or limited communication and heterogeneous data distributions. Two notable trends to deal with the communication overhead of federated algorithms are *gradient compression* and *local computation with periodic communication*. Despite many attempts, characterizing the relationship between these two approaches has proven elusive. We address this by proposing a set of algorithms with periodical compressed (quantized or sparsified) communication and analyze their convergence properties in both homogeneous and heterogeneous local data distributions settings. For the homogeneous setting, our analysis improves existing bounds by providing tighter convergence rates for both *strongly convex* and *non-convex* objective functions. To mitigate data heterogeneity, we introduce a *local gradient tracking* scheme and obtain sharp convergence rates that match the best-known communication complexities without compression for convex, strongly convex, and nonconvex settings. We complement our theoretical results by demonstrating the effectiveness of our proposed methods on real-world datasets. The code is available at https://github.com/MLOPTPSU/FedTorch.

1 Introduction

The primary obstacle towards scaling distributed optimization algorithms is the significant communication cost both in terms of the number of communication rounds and the amount of exchanged data per round. To significantly reduce the number of communication rounds, a practical solution is to trade-off local computation for less communication via periodic averaging [55, 69]. In particular, the local SGD algorithm [55, 60, 67] alternates between a fixed number of local updates and one step of synchronization which is shown to enjoy the same convergence rate as its fully synchronous counterpart, while significantly reducing the number of communication rounds.

A fundamentally different solution to scale up distributed optimization algorithms is to reduce the size of the communicated message per communication round. This problem is especially exacerbated in edge computing where the worker devices (e.g., smartphones or IoT devices) are remotely connected, and communication bandwidth and power resources are limited. For instance, ResNet [20] has more than 25 million parameters, so the communication cost of sending local models through a computer network could be prohibitive. The current methodology towards reducing the size of messages is to communicate compressed local gradients or models to the central server by utilizing a quantization operator [3, 6, 48, 58, 59, 63, 65], sparsification schema [4, 39, 56, 65], or composition of both [5].

Despite significant progress in improving both aspects of communication efficiency [6, 26, 56, 57], there still exists a huge gap in our understanding of these approaches in federated learning, in particular for the cases that both compression and periodic averaging techniques are applied simultaneously. In terms of reducing communication rounds, a few recent attempts were able to reduce the frequency of synchronizing locally evolving models [14, 27], which are not improvable in general [70]. This necessitates that further improvement in communication efficiency needs to be explored by reducing the size of communicated messages. We

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

	Objective function		
Reference	Nonconvex	PL/Strongly Convex	General Convex
QSPARSE [5]	$R=O\left(\frac{q+1}{\epsilon^{3/2}}\right)$ $\tau=O\left(\frac{1}{m(q+1)\sqrt{\epsilon}}\right)$	$R=O\left(\kappa\frac{q+1}{\sqrt{\epsilon}}\right)$ $\tau=O\left(\frac{1}{m(q+1)\sqrt{\epsilon}}\right)$	–
FedPAQ [48]	$R=O\left(\frac{1}{\epsilon}\right)$ $\tau=O\left(\frac{(\frac{q}{m})^2+1}{\epsilon}\right)$	$R=O\left(m+\frac{q+1}{m\epsilon}\right)$ $\tau=O\left(1\right)$	–
Theorem 5.1	$\boldsymbol{R=O\left(\frac{1}{\epsilon}\right)}$ $\boldsymbol{\tau=O\left(\frac{q+1}{m\epsilon}\right)}$	$\boldsymbol{R=O\left(\kappa\left(\frac{q}{m}+1\right)\log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau=O\left(\frac{(q+1)}{m\left(\frac{q}{m}+1\right)\epsilon}\right)}$	$\boldsymbol{R=O\left(\frac{1+\frac{q}{m}}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau=O\left(\frac{(q+1)^2}{m\left(\frac{q}{m}+1\right)^2\epsilon^2}\right)}$

Table 1: Comparison of results with compression and periodic averaging in the homogeneous setting. Here, m is the number of devices, q is compression distortion constant, κ is condition number, ϵ is target accuracy, R is the number of communication rounds, and τ is the number of local updates. QSPARSE [5] has the assumption of bounded gradient, while FedPAQ [48] and our proposed algorithm do not have such assumption.

highlight that compressed communication is of further importance to accelerate training non-convex objectives as it requires significantly more communication rounds to converge compared to distributed convex optimization. Furthermore, most existing methods are analyzed for homogeneous data and our understanding of the efficiency of these methods in the heterogeneous case is lacking.

In light of the above issues, the key contribution of this paper is the introduction and analysis of simple variants of *local SGDwith compressed communication* without compromising the attainable guarantees. The proposed algorithmic ideas accommodate both homogeneous and heterogeneous data distributions settings with the obtained rates summarized in Table 1 and Table 2, respectively. In the homogeneous case, with a tight analysis of a simple quantized variant of local SGD, we show that not only our proposed method improves the complexity bounds for algorithms with compression (Table 1), but also outperforms the complexity bounds for non-compressed counterparts in terms of the number of communication rounds (Table 3). In the heterogeneous case, we argue that in the presence of compression (quantization or sparsification), locally updating models via local gradient information could lead to a significant drift among local models, which shed light on designing a quantized variant of local SGD that tracks local gradient information at local devices. We show that this simple gradient tracking idea leads to a method that outperforms state-of-the-art methods with compression for the heterogeneous setting (Table 2) and it can even compensate for the noise introduced by compression and lead to the best-known convergence rates for convex and non-convex settings under perfect communication, i.e., no compression (Table 4).

	Objective function		
Reference	Nonconvex	PL/Strongly Convex	General Convex
QSPARSE [5]	$R=O\left(\frac{q+1}{\epsilon^{3/2}}\right)$ $\tau=O\left(\frac{1}{m(q+1)\sqrt{\epsilon}}\right)$	$R=O\left(\kappa\frac{q+1}{\sqrt{\epsilon}}\right)$ $\tau=O\left(\frac{1}{m(q+1)\sqrt{\epsilon}}\right)$	–
Theorem 5.2	$\boldsymbol{R=O\left(\frac{q+1}{\epsilon}\right)}$ $\boldsymbol{\tau=O\left(\frac{1}{m\epsilon}\right)}$	$\boldsymbol{R=O\left(\kappa\left(q+1\right)\log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau=O\left(\frac{1}{m\epsilon}\right)}$	$\boldsymbol{R=O\left(\frac{1+q}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau=O\left(\frac{1}{m\epsilon^2}\right)}$

Table 2: Comparison of results with compression and periodic averaging in the heterogeneous setting. QSPARSE [5] has the assumption of bounded gradient, while our proposed algorithm does not.

Contributions. We summarize the main contributions of this paper below:

- *Homogeneous local distributions:* To keep the analysis simple yet insightful, we start with a quantized variant of federated averaging algorithm and analyze its convergence for non-convex, strongly convex and general convex objectives. As demonstrated in Table 1, the obtained rates is novel for convex objectives to the best of our knowledge, and improves the best known bounds in [48] and [5] for general non-convex and strongly convex objectives, respectively.

- *Heterogeneous local distributions:* For the heterogeneous setting, we propose federated averaging with compression and local gradient tracking, dubbed as `FedCOMGATE` algorithm, and establish its convergence rates for general non-convex, strongly convex or PL, and convex objectives. The obtained rates improve upon the results reported in [5] for general non-convex and strongly-convex objectives. The obtained rates for general convex functions are novel to the best of our knowledge.

- We verify our theoretical results through various extensive experiments on different real federated datasets that demonstrate the practical efficacy of our methods.

2 Problem Setup

In this paper we focus on a federated architecture, where m users aim to learn a global model in a collaborative manner without exchanging their data points with each other. Moreover, we assume that users (computing units) can only exchange information via a central unit (server) which is connected to all users. The optimization problem that the users try to solve can be written as

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) \triangleq \frac{1}{m}\sum_{j=1}^{m} f_j(\boldsymbol{w}) \qquad (1)$$

where $f_j : \mathbb{R}^d \rightarrow \mathbb{R}$ is the loss function corresponding to user j. We further assume that the local objective function of each user j is the expected loss over the set of data points of node j, i.e.,

$$f_j(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{P}_j}[\ell_j(\boldsymbol{w},\boldsymbol{z})], \tag{2}$$

where $\boldsymbol{z}$ is a random variable with probability distribution $\mathcal{P}_j$ and the loss function ℓ_j measures how well the model performs. $\mathcal{P}_j$ can be considered as the underlying distribution of node j for generating data points, and realizations of the random variable $\boldsymbol{z}$ are the data points of node j. For instance, in a supervised learning case each element sample point $\boldsymbol{z}_i$ corresponds to a pair of input (feature) vector $\boldsymbol{x}_i$ and its label y_i. In this case, $\ell_j(\boldsymbol{w},\boldsymbol{z}_i) = \ell_j(\boldsymbol{w},(\boldsymbol{x}_i,y_i))$ measures how well the model $\boldsymbol{w}$ performs in predicting the label of $\boldsymbol{x}_i$ which is y_i. Note that the probability distributions of users may not be necessarily identical. In fact, through the paper, we study two settings (i) homogeneous setting in which all the probability distributions and loss functions are identical, i.e., $(\mathcal{P}_1 = \ldots = \mathcal{P}_m)$ and $(\ell_1 = \ldots = \ell_m)$; and (ii) heterogeneous setting in which the users' distributions and loss functions could be different.

3 Federated Averaging with Compression[1]

In this section, we propose a generalized version of the local stochastic gradient descent (SGD) method for federated learning which uses compressed signals to reduce the overall communication overhead of solving problem (1). The proposed federated averaging with compression (`FedCOM`) is designed for homogeneous settings where the probability distributions and loss functions of the users are identical. `FedCOM` differs from standard local SGD methods [55, 60, 67] in two major aspects. First, it uses compressed messages for uplink communication. Second, at the central node, the new global model is a convex combination of the previous global model and the average of updated local models of users. We show that `FedCOM` converges faster than state-of-the-art methods in a homogeneous setting by periodic averaging, local and global learning rates, and compressed communications.

To formally present the steps of `FedCOM`, consider R as the rounds of communication between server and users, and τ as the number of local updates performed between two consecutive communication rounds. Further, define $\boldsymbol{w}^{(r)}$ as the model at the master at the r-th round of communication. At each round r, the server sends the global model $\boldsymbol{w}^{(r)}$ to the users (clients). Then, each user j computes its local stochastic gradient and updates the model by following the update of SGD for τ iterations. Specifically, at communication round r, user j follows the update

Algorithm 1: `FedCOM`$(R,\ \tau,\eta,\gamma)$

```
Inputs: Number of communication rounds $R$,
  number of local updates $\tau$, learning rates $\gamma$ and $\eta$,
  initial global model $\boldsymbol{w}^{(0)}$
for $r = 0,\ldots,R-1$ do
    for each client $j \in [m]$ do in parallel
        Set $\boldsymbol{w}_j^{(0,r)} = \boldsymbol{w}^{(r)}$
        for $c = 0,\ldots,\tau-1$ do
            Sample a minibatch $\mathcal{Z}_j^{(c,r)}$ and compute
              $\tilde{\mathbf{g}}_j^{(c,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(c,r)};\mathcal{Z}_j^{(c,r)})$
            $\boldsymbol{w}_j^{(c+1,r)} = \boldsymbol{w}_j^{(c,r)} - \eta\,\tilde{\mathbf{g}}_j^{(c,r)}$
        end
        Device sends $\boldsymbol{\Delta}_{j,q}^{(r)} = Q((\boldsymbol{w}^{(r)} - \boldsymbol{w}_j^{(\tau,r)})/\eta)$
          back to the server
    end
    Server computes $\boldsymbol{\Delta}_q^{(r)} = \frac{1}{m}\sum_{j=1}^{m}\boldsymbol{\Delta}_{j,q}^{(r)}$
    Server computes $\boldsymbol{w}^{(r+1)} = \boldsymbol{w}^{(r)} - \eta\gamma\boldsymbol{\Delta}_q^{(r)}$ and
      broadcasts to all devices
end
```

$$\boldsymbol{w}_j^{(c+1,r)} = \boldsymbol{w}_j^{(c,r)} - \eta\,\tilde{\mathbf{g}}_j^{(c,r)}, \qquad \text{for} \quad c = 0,\ldots,\tau-1. \tag{3}$$

Here, $\boldsymbol{w}_j^{(c,r)}$ is the model at node j and round r after c local updates, $\tilde{\mathbf{g}}_j^{(c,r)} := \nabla f_j(\boldsymbol{w}_j^{(c,r)};\mathcal{Z}_j^{(c,r)}) := \frac{1}{b_j}\sum_{\mathbf{z}\in\mathcal{Z}_j^{(c,r)}}\nabla\ell_j(\boldsymbol{w}_j^{(c,r)},\mathbf{z})$ is a stochastic gradient of f_j evaluated using the mini-batch $\mathcal{Z}_j^{(c,r)} := \{\boldsymbol{z}_{j,1}^{(c,r)},\ldots,\boldsymbol{z}_{j,b_j}^{(c,r)}\}$ of size b_j, and η is the learning rate. The output of this τ recursive updates for node j at round r is $\boldsymbol{w}_j^{(\tau,r)}$. After computing the local models, each user j sends a compressed version of $(\boldsymbol{w}_j^{(\tau,r)} - \boldsymbol{w}_j^{(r)})/\eta$ to the central node by applying a compression operator $Q(\cdot)$. Note that the compressed signal $\boldsymbol{\Delta}_{j,q}^{(r)} \triangleq Q((\boldsymbol{w}_j^{(\tau,r)} - \boldsymbol{w}_j^{(r)})/\eta)$ indicates a normalized version of the difference between the input and output of the local SGD process at round r at node j, which is equal to the aggregation of all local SGD directions, i.e., $(\boldsymbol{w}_j^{(\tau,r)} - \boldsymbol{w}_j^{(r)})/\eta = \sum_{c=0}^{\tau}\tilde{\mathbf{g}}_j^{(c,r)}$. Once, the server receives the compressed signals $\{\boldsymbol{\Delta}_{j,q}^{(r)}\}_{j=1}^{j=m}$, it computes the new global model according to

$$\boldsymbol{w}^{(r+1)} = \boldsymbol{w}^{(r)} - \frac{\eta\gamma}{m}\sum_{j=1}^{m}\boldsymbol{\Delta}_{j,q}^{(r)}, \tag{4}$$

where γ is the global learning rate. The steps of the `FedCOM` algorithm are summarized in Algorithm 1.

Remark 1. *Note that by setting $\gamma = 1$ in (4), `FedCOM` boils down to the FedPAQ algorithm proposed in [48], and if we further remove the compression scheme then we recover FedAvg [60]. Note that in both FedAvg and its vanilla quantized variant FedPAQ, the new global*

[1] Generalized Compressed Local SGD

model is the average of local models (if we ignore the error of compression for FedPAQ), while in `FedCOM` *the new global model is a linear combination of the previous global model and the average of updated local models, due to the extra parameter γ. We show that by adding this modification and properly choosing γ,* `FedCOM` *improves the complexity bounds of FedPAQ for both strongly convex and non-convex settings. Note that the update in (4) can also be interpreted as running a global SGD update on master's model by descending towards the average of aggregated local gradient directions with stepsize $\eta\gamma$. Specifically, if we assume perfect communication (ignoring the quantization) then we obtain that the new global model is given by $\boldsymbol{w}^{(r+1)} = \boldsymbol{w}^{(r)} - \eta\gamma\frac{1}{m}\sum_{j=1}^{m}\sum_{c=0}^{\tau}\tilde{\mathbf{g}}_j^{(c,r)}$.*

4 Compressed Local SGD with Local Gradient Tracking

In the previous section, we introduced a relatively simple algorithm called `FedCOM` for homogeneous settings, where the probability distributions of the users are identical. Although `FedCOM` both theoretically (Section 5) and numerically (Section 6) performs well for homogeneous settings, its performance is not satisfactory in heterogeneous settings where the probability distributions of users are different. This is due to the fact that the updates of `FedCOM` heavily depend on the local SGD directions. In a homogeneous setting, following local gradient directions leads to a good global model as all samples are drawn from the same distribution and the local gradient direction is a good estimate of the global function gradient. However, in a heterogeneous setting, updating local models only based on local gradient information could lead to an arbitrary poor performance as the local gradient directions could be very different from the global gradient direction.

To address this issue, in this section we propose a novel variant of federated averaging with compression and local gradient tracking (`FedCOMGATE`) for heterogeneous settings. The main difference between `FedCOM` and `FedCOMGATE` is the idea of local gradient tracking that ensures that each node uses an estimate of the global gradient direction to locally update its model. To estimate global gradient direction nodes also require access to the average of local models which means that in `FedCOMGATE` in addition to sending the global updates master also needs to broadcast the average of $\boldsymbol{\Delta}_{j,q}^{(r)} \triangleq Q((\boldsymbol{w}_j^{(\tau,r)} - \boldsymbol{w}_j^{(r)})/\eta)$, shown by $\boldsymbol{\Delta}_q^{(r)} = \frac{1}{m}\sum_{j=1}^{m}\boldsymbol{\Delta}_{j,q}^{(r)}$ to devices.

To present `FedCOMGATE`, consider δ_j as a sequence at node j that is designed to track the difference between the local gradient direction and the global gradient direction (the direction obtained by incorporating gradient information of all users). At round r, each worker j updates its local sequence δ_j based on the update

$$\delta_j^{(r+1)} = \delta_j^{(r)} + \frac{1}{\tau}\left(\boldsymbol{\Delta}_{j,q}^{(r)} - \boldsymbol{\Delta}_q^{(r)}\right), \qquad (5)$$

where $\boldsymbol{\Delta}_{j,q}^{(r)}$ is the quantized version of the accumulation of the gradients at node j from the previous round and $\boldsymbol{\Delta}_q^{(r)}$ is the average of $\boldsymbol{\Delta}_{j,q}^{(r)}$. Once the correction vector $\delta_j^{(r)}$ is computed, each node j runs a corrected local update for τ rounds based on the update

$$\boldsymbol{w}_j^{(c+1,r)} = \boldsymbol{w}_j^{(c,r)} - \eta\, \tilde{\boldsymbol{d}}_{j,q}^{(c,r)} = \boldsymbol{w}_j^{(c,r)} - \eta(\tilde{\mathbf{g}}_j^{(c,r)} - \delta_j^{(r)}), \quad \text{for} \quad c = 0, \ldots, \tau-1, \quad (6)$$

where $\tilde{\mathbf{g}}_j^{(c,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(c,r)}\mathcal{Z}_j^{(c,r)})$ is the stochastic gradient of node j at round r for the c-th local update. In the above update the local descent direction $\tilde{\boldsymbol{d}}_{j,q}^{(c,r)}$ is defined as the difference the local stochastic gradient $\tilde{\mathbf{g}}_j^{(c,r)}$ and the correction vector $\delta_j^{(r)}$ which aims to track the difference between local and global gradient directions. Note that for all τ local updates at round r, the vector $\delta_j^{(r)}$ is fixed while the local stochastic gradient $\tilde{\mathbf{g}}_j^{(c,r)}$ is computed via fresh samples for each local update. Once the local models $\boldsymbol{w}_j^{(\tau,r)}$ are computed, nodes send their quantized accumulation of gradients $\boldsymbol{\Delta}_{j,q}^{(r)}$ to the server. Then, the server uses this information to compute the average update $\boldsymbol{\Delta}_q^{(r)} \triangleq \frac{1}{m}\sum_{j=1}^{m}\boldsymbol{\Delta}_{j,q}^{(r)}$ and broadcasts it to the devices. Moreover, the server utilizes $\boldsymbol{\Delta}_q^{(r)}$ to compute the new global model $\boldsymbol{w}^{(r+1)}$ according to (4). The steps of `FedCOMGATE` are outlined in Algorithm 2.

Comparison with SCAFFOLD [26] and VRL-SGD in [36]. From an algorithmic standpoint, in comparison to the SCAFFOLD method proposed in [26] , in addition to the fact that we use compressed signals to further reduce the communication overhead, we would like to highlight that our algorithm is much simpler and does not require any extra control variable (see Eq. (4) and Eq. (5) in [26] for more details). Also, since we do not use an extra control variable, the extension of our convergence analysis to the case where a subset of devices participate at each communication round is straightforward and for clarity, we do not include analysis with device sampling. Yet, we shall study the impact of device sampling empirically (see Figure 5 in Section 6 and Algorithm 4 in Appendix B.1). In comparison to [36] which employs an explicit variance reduction component, if we let $Q(\boldsymbol{x}) = \boldsymbol{x}$ (case of no quantization), our algorithm reduces to a generalization of algorithm in [36] with distinct local and global learning rates. We note that for the case of $\gamma = 1$

Algorithm 2: FedCOMGATE(R, τ, η, γ)

```
Inputs: Number of communication rounds $R$,
  number of local updates $\tau$, learning rates $\gamma$ and $\eta$,
  initial global model $\boldsymbol{w}^{(0)}$, initial gradient tracking
  $\delta_j^{(0)} = \mathbf{0}$, $\forall j \in [m]$
for $r = 0, \ldots, R-1$ do
    for each client $j \in [m]$ do in parallel
        Set $\boldsymbol{w}_j^{(0,r)} = \boldsymbol{w}^{(r)}$
        for $c = 0, \ldots, \tau-1$ do
            Set $\tilde{\boldsymbol{d}}_{j,q}^{(c,r)} = \tilde{\mathbf{g}}_j^{(c,r)} - \delta_j^{(r)}$ where
              $\tilde{\mathbf{g}}_j^{(c,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(c,r)}; \mathcal{Z}_j^{(c,r)})$
            $\boldsymbol{w}_j^{(c+1,r)} = \boldsymbol{w}_j^{(c,r)} - \eta\, \tilde{\boldsymbol{d}}_{j,q}^{(c,r)}$
        end
        Device sends $\boldsymbol{\Delta}_{j,q}^{(r)} = Q((\boldsymbol{w}^{(r)} - \boldsymbol{w}_j^{(\tau,r)})/\eta)$
          to the server
        Device updates
          $\delta_j^{(r+1)} = \delta_j^{(r)} + \frac{1}{\tau}(\boldsymbol{\Delta}_{j,q}^{(r)} - \boldsymbol{\Delta}_q^{(r)})$
    end
    Server computes $\boldsymbol{\Delta}_q^{(r)} = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{\Delta}_{j,q}^{(r)}$ and
      broadcasts back to all devices
    Server computes $\boldsymbol{w}^{(r+1)} = \boldsymbol{w}^{(r)} - \eta\gamma\boldsymbol{\Delta}_q^{(r)}$ and
      broadcasts to all devices
end
```

and $Q(\boldsymbol{x}) = \boldsymbol{x}$ the FedCOMGATE$(\tau, \eta, \gamma = 1)$ reduces to the federated algorithm proposed in [36] with minor distinction that our algorithm's output is the global model at the server.

5 Convergence Analysis

Next, we present the convergence analysis of our proposed methods. First, we state our assumptions.

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

Assumption 2. *The output of the compression operator $Q(\boldsymbol{x})$ is an unbisased estimator of its input $\boldsymbol{x}$, and and its variance grows with the squared of the squared of ℓ_2-norm of its argument, i.e., $\mathbb{E}[Q(\boldsymbol{x})|\boldsymbol{x}] = \boldsymbol{x}$ and $\mathbb{E}[\|Q(\boldsymbol{x}) - \boldsymbol{x}\|^2|\boldsymbol{x}] \leq q\, \|\boldsymbol{x}\|^2$.*

Assumptions 1-2 are customary in the analysis of methods with compression, and they will all be assumed in all of our results. We should also add that several quantization approaches and sparsification techniques satisfy the condition in Assumption 2. For examples of such compression schemes we refer the reader to [5, 21]. We report our results for three different class of loss functions: (i) nonconvex (ii) convex (iii) non-convex Polyak-Łojasiewicz (PL). Indeed, as any μ-strongly convex is μ-PL [25], our results for the PL case automatically hold for strongly convex functions.

5.1 Convergence of FedCOM in the homogeneous data distribution setting

Now we focus on the homogeneous case in which the stochastic local gradient of each worker is an unbiased estimator of the global gradient.

Assumption 3 (Bounded Variance)**.** *For all $j \in [m]$, we can sample an independent mini-batch $\mathcal{Z}_j$ of size $|\mathcal{Z}_j^{(c,r)}| = b$ and compute an unbiased stochastic gradient $\tilde{\mathbf{g}}_j = \nabla f_j(\boldsymbol{w}; \mathcal{Z}_j), \mathbb{E}_{\mathcal{Z}_j}[\tilde{\mathbf{g}}_j] = \nabla f(\boldsymbol{w}) = \mathbf{g}$. Moreover, their variance is bounded above by a constant σ^2, i.e., $\mathbb{E}_{\mathcal{Z}_j}\left[\|\tilde{\mathbf{g}}_j - \mathbf{g}\|^2\right] \leq \sigma^2$.*

In the following theorem, we state our main theoretical results for FedCOM in the homogeneous setting.

Theorem 5.1. *Consider FedCOM in Algorithm 1. Suppose that the conditions in Assumptions 1-3 hold. If the local data distributions of all users are identical (homogeneous setting), then we have*

- ***Nonconvex:*** *By choosing stepsizes as $\eta = \frac{1}{L\gamma}\sqrt{\frac{m}{R\tau(q+1)}}$ and $\gamma \geq m$, the sequence of iterates satisfies $\frac{1}{R}\sum_{r=0}^{R-1}\left\|\nabla f(\boldsymbol{w}^{(r)})\right\|_2^2 \leq \epsilon$ if we set $R = O\left(\frac{1}{\epsilon}\right)$ and $\tau = O\left(\frac{q+1}{m\epsilon}\right)$.*
- ***Strongly convex or PL:*** *By choosing stepsizes as $\eta = \frac{1}{2L\left(\frac{q}{m}+1\right)\tau\gamma}$ and $\gamma \geq m$, we obtain the iterates satisfy $\mathbb{E}\Big[f(\boldsymbol{w}^{(R)}) - f(\boldsymbol{w}^{(*)})\Big] \leq \epsilon$ if we set $R = O\left(\left(\frac{q}{m}+1\right)\kappa\log\left(\frac{1}{\epsilon}\right)\right)$ and $\tau = O\left(\frac{q+1}{m\left(\frac{q}{m}+1\right)\epsilon}\right)$.*
- ***Convex:*** *By choosing stepsizes as $\eta = \frac{1}{2L\left(\frac{q}{m}+1\right)\tau\gamma}$ and $\gamma \geq m$, we obtain that the iterates satisfy $\mathbb{E}\Big[f(\boldsymbol{w}^{(R)}) - f(\boldsymbol{w}^{(*)})\Big] \leq \epsilon$ if we set $R = O\left(\frac{L\left(1+\frac{q}{m}\right)}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)$ and $\tau = O\left(\frac{(q+1)^2}{m\left(\frac{q}{m}+1\right)^2\epsilon^2}\right)$.*

Theorem 5.1 characterizes the number of required local updates τ and communication rounds R to achieve an ϵ-first-order stationary point for the nonconvex setting and an ϵ-suboptimal solution for convex and strongly convex settings, when we are in a homogeneous case. A few important observations follow. First, in all three results the dependency of τ and R on the variance of compression scheme q is scaled down by a factor of $1/m$. Hence, by cooperative learning the users are able to lower the effect of the noise induced by the compression scheme. Second, in all three cases, the number of local updates τ required for achieving a specific accuracy is

Reference	Objective function		
	Nonconvex	PL/Strongly Convex	General Convex
Local-SGD [14]	–	$R = O\left(\left(\frac{1}{\epsilon}\right)^{\frac{1}{3}}\right)$ $\tau = O\left(\frac{1}{m\epsilon}\right)$	–
Local-SGD [27]	–	$R = O\left(m\kappa \log\left(\frac{1}{\epsilon}\right)\right)$ $\tau = O\left(\frac{1}{m^2\epsilon}\right)$	$R = O\left(\frac{m}{\epsilon}\right)$ $\tau = O\left(\frac{1}{m^2\epsilon}\right)$
Local-SGD [60]	$R = O\left(\frac{m}{\epsilon}\right)$ $\tau = O\left(\frac{1}{m^2\epsilon}\right)$	–	–
Theorem 5.1	$\boldsymbol{R = O\left(\frac{1}{\epsilon}\right)}$ $\boldsymbol{\tau = O\left(\frac{1}{m\epsilon}\right)}$	$\boldsymbol{R = O\left(\kappa \log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau = O\left(\frac{1}{m\epsilon}\right)}$	$\boldsymbol{R = O\left(\frac{1}{\epsilon} \log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau = O\left(\frac{1}{m\epsilon^2}\right)}$

Table 3: Comparison of `FedCOM` with results that use periodic averaging but do not utilize compression, i.e., $q = 0$, in the homogeneous setting.

proportional to $1/m$. In the homogeneous setting, this result is expected since we have m machines and the number of samples used per local update is m times of the case that only a single machine runs local SGD. As a result, the overall number of required local updates scales inversely by the number of machines m. Third, in all three cases, the dependency of communication rounds R on the required accuracy ϵ matches the number of required updates for solving that problem in centralized deterministic settings. For instance, in a centralized nonconvex setting, to achieve a point that satisfies $\|\nabla f(\boldsymbol{w})\|^2 \leq \epsilon$ we need $O(1/\epsilon)$ gradient updates for deterministic case and $O(1/\epsilon^2)$ SGD updates for the stochastic case. It is interesting that running $\tau = O(1/\epsilon)$ local updates controls the noise of stochastic gradients and the number of communication rounds $R = O(1/\epsilon)$ stays same as the centralized deterministic case. Similar observations hold for convex (upto a log factor) and strongly convex cases.

Remark 2. *While the bound obtained in Theorem 5.1 for general non-convex objectives indicates that achieving a convergence rate of ϵ requires $R = O\left(\frac{q+1}{\epsilon}\right)$ communication rounds with $\tau = O\left(\frac{1}{m\epsilon}\right)$ local updates, in Remark 7 in Appendix D, we show that the same rate can be achieved with $R = O\left(\frac{1}{\epsilon}\right)$ and $\tau = O\left(\frac{q+1}{m\epsilon}\right)$. Hence, the noise of quantiziation can be compensated with higher number of local steps τ.*

Remark 3. *The results for `FedCOM` improve the complexity bounds for other federated leaning methods with compression (in the homogeneous setting) that are proposed in [48] and [5]. Check Table 1 for more details.*

Remark 4. *To show the tightness of our result for `FedCOM`, we also compare its results with schemes without compression, $q = 0$, developed for homogeneous settings, shown in Table 3. As we observe, the number of required communication rounds for `FedCOM` without compression in convex, strongly convex, and nonconvex settings are smaller than the best-known rates for each setting by a factor of $\frac{1}{m}$.*

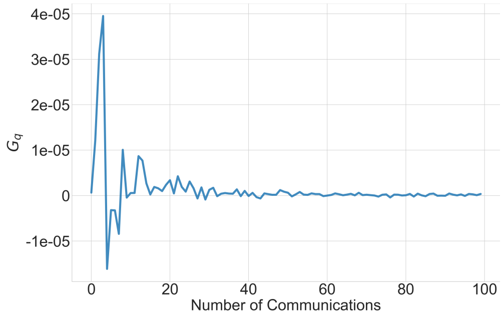


Figure 1: The error of quantization measured by Assumption 5 for `FedCOMGATE` with the MNIST dataset applied to a MLP model. We quantized updates $\boldsymbol{\Delta}_j$s from 32 bits floating-point to 8 bits integer.

5.2 Convergence of `FedCOMGATE` in the data heterogeneous setting

Next, we report our results for `FedCOMGATE` in the heterogeneous setting. We consider a less strict assumption compared to Assumption 3, that the stochastic gradient of each user is an unbiased estimator of its local gradient with bounded variance.

Assumption 4 (Bounded Variance)**.** *For all $j \in [m]$, we can sample an independent mini-batch $\mathcal{Z}_j$ of size $|\mathcal{Z}_j| = b$ and compute an unbiased stochastic gradient $\tilde{\mathbf{g}}_j = \nabla f_j(\boldsymbol{w}; \mathcal{Z}_j), \mathbb{E}_\xi[\tilde{\mathbf{g}}_j] = \nabla f_j(\boldsymbol{w}) = \mathbf{g}_j$. Moreover, the variance of local stochastic gradients is bounded above by a constant σ^2, i.e., $\mathbb{E}_\xi\left[\|\tilde{\mathbf{g}}_j - \mathbf{g}_j\|^2\right] \leq \sigma^2$.*

Assumption 5. *The compression scheme Q for the heterogeneous data distribution setting satisfies the following condition $\mathbb{E}_Q[\|\frac{1}{m}\sum_{j=1}^m Q(\boldsymbol{x}_j)\|^2 - \|Q(\frac{1}{m}\sum_{j=1}^m \boldsymbol{x}_j)\|^2] \leq G_q$.*

The condition in Assumption 4 is not strict and only ensures that the local stochastic gradients are unbiased estimators of local gradients with bounded variance. Regarding Assumption 5, for the case of no compression, $Q(\boldsymbol{x}) = \boldsymbol{x}$, the compression error becomes naturally $G_q = 0$. We highlight that this assumption is only needed in the heterogeneous setting, and since both of the terms in the argument of expectation depend on the quantization, this assumption can be seen as a weaker version of the gradient diversity assumptions in the convergence analysis of heterogeneous settings. To show how this assumption holds in practice, refer to Figure 1 in Appendix B.2. we run an experiment on the MNIST dataset using `FedCOMGATE` algorithm with quantizing gradients from 32 bits floating-point to 8 bits integer. In Figure 1 we plot changes in G_q quantity through this experiment. It shows that G_q is decreasing as we proceed with the training, simply because the ℓ_2-norm of the updated vector is going to zero. Note that the quantity of G_q could be even

	Objective function		
Reference	Nonconvex	PL/Strongly Convex	General Convex
SCAFFOLD [26]	$R = O\left(\frac{1}{\epsilon}\right)$ $\tau = O\left(\frac{1}{m\epsilon}\right)$	$R = O\left(\kappa \log\left(\frac{1}{\epsilon}\right)\right)$ $\tau = O\left(\frac{1}{m\epsilon}\right)$	$R = O\left(\frac{1}{\epsilon}\right)$ $\tau = O\left(\frac{1}{m\epsilon}\right)$
Local-SGD [27]	–	–	$R = O\left(\frac{1}{\epsilon^{1.5}}\right)$ $\tau = O\left(\frac{1}{m\epsilon^{0.5}}\right)$
VRL-SGD [36]	$R = O\left(\frac{m}{\epsilon}\right)$ $\tau = O\left(\frac{1}{m^2\epsilon}\right)$	–	–
Theorem 5.2	$\boldsymbol{R = O\left(\frac{1}{\epsilon}\right)}$ $\boldsymbol{\tau = O\left(\frac{1}{m\epsilon}\right)}$	$\boldsymbol{R = O\left(\kappa \log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau = O\left(\frac{1}{m\epsilon}\right)}$	$\boldsymbol{R = O\left(\frac{1}{\epsilon} \log\left(\frac{1}{\epsilon}\right)\right)}$ $\boldsymbol{\tau = O\left(\frac{1}{m\epsilon^2}\right)}$

Table 4: Comparison of `FedCOMGATE` with results that use periodic averaging but do not utilize compression, i.e., $q = 0$, in the heterogeneous setting. While SCAFFOLD [26] requires to communicate 2 vectors to the server in the uplink, other algorithms only communicate 1 vector.

negative as illustrated in Figure 1. For more details on the experiment see Section 6.

Next, we present our main theoretical results for the `FedCOMGATE` method in the heterogeneous setting.

Theorem 5.2. *Consider `FedCOMGATE` in Algorithm 2. If Assumptions 1, 2, 4 and 5 hold, then even for the case the local data distribution of users are different (heterogeneous setting) we have*

- ***Non-convex:*** *By choosing stepsizes as* $\eta = \frac{1}{L\gamma}\sqrt{\frac{m}{R\tau(q+1)}}$ *and* $\gamma \geq m$*, we obtain that the iterates satisfy* $\frac{1}{R}\sum_{r=0}^{R-1}\left\|\nabla f(\boldsymbol{w}^{(r)})\right\|_2^2 \leq \epsilon$ *if we set* $R = O\left(\frac{q+1}{\epsilon}\right)$ *and* $\tau = O\left(\frac{1}{m\epsilon}\right)$.
- ***Strongly convex or PL:*** *By choosing stepsizes as* $\eta = \frac{1}{2L(\frac{q}{m}+1)\tau\gamma}$ *and* $\gamma \geq \sqrt{m\tau}$*, we obtain that the iterates satisfy* $\mathbb{E}\Big[f(\boldsymbol{w}^{(R)}) - f(\boldsymbol{w}^{(*)})\Big] \leq \epsilon$ *if we set* $R = O\left((q+1)\,\kappa \log\left(\frac{1}{\epsilon}\right)\right)$ *and* $\tau = O\left(\frac{1}{m\epsilon}\right)$.
- ***Convex:*** *By choosing stepsizes as* $\eta = \frac{1}{2L(q+1)\tau\gamma}$ *and* $\gamma \geq \sqrt{m\tau}$*, we obtain that the iterates satisfy* $\mathbb{E}\Big[f(\boldsymbol{w}^{(R)}) - f(\boldsymbol{w}^{(*)})\Big] \leq \epsilon$ *if we set* $R = O\left(\frac{L(1+q)}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)$ *and* $\tau = O\left(\frac{1}{m\epsilon^2}\right)$.

The implications of Theorem 5.2 are similar to the ones for Theorem 5.1. Yet, unlike the homogeneous setting, the compression variance q does not scale down by a factor of $1/m$. We emphasize that similar to the homogeneous case, in all three cases, the dependency of R on ϵ matches the number of required update for solving the problem in centralized fashion.

Remark 5. *To show the tightness of our result for `FedCOMGATE`, we also compare its complexity bounds with other schemes without compression,* $q = 0$*, for heterogeneous settings, summarized in Table 4. As it can be observed in all settings, the bound for `FedCOMGATE` without compression (`FedGATE`) matches the best-known complexity bounds for these settings (upto a log factor).*

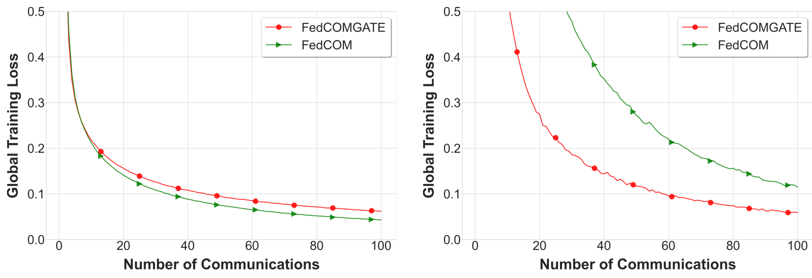
Remark 6. *We highlight that since we do not need to communicate control variate in uplink, the communication cost of our algorithm is half of the corresponding cost in SCAFFOLD. Yet, for downlink communication, similar to SCAFFOLD our communication cost is doubled compared to FedAvg due to gradient tracking. However, we emphasize that in general communication cost of broadcasting a message is much cheaper than uplink communication.*

6 Experiments

In this section, we empirically validate the performance of proposed algorithms. We compare our methods with FedAvg [43], its quantized version, FedPAQ [48], and SCAFFOLD [26] for heterogeneous federated learning. In addition, we present a variant of our algorithm without compression dubbed as `FedGATE`. The details of `FedGATE` is described in Algorithm 3 in Appendix B.1. Also, a variant of our algorithm with client sampling is presented in Algorithm 4 in Appendix B.1. In addition to what is presented here, in Appendix B.2, we explore the effects of client sampling, local computation, quantization and sparsification on the convergence of our proposed algorithms.

Setup. We implement our algorithms on the `Distributed` library of `PyTorch` [45], using Message Passing Interface (`MPI`), in order to simulate the real-world collaborative learning scenarios such as the one in federated learning. We run the experiments on a HPC cluster with 3 `Intel Xeon E5-2695` CPUs, each of which with 28 processes. For this experiment we use four main datasets: `MNIST` [1], `CIFAR10` [31], `Fashion MNIST` [66] and `EMNIST` [9]. For each experiment, we have 100 devices communicating with the server. Each experiment runs for 100 rounds of communication between clients and the server, and we report the global model loss on the training data averaged over all clients and the test accuracy over the global model. For `MNIST` and `Fashion MNIST` we use an MLP model with two hidden layers, each with 200 neurons with ReLU activations. For the `CIFAR10` dataset, we use the same MLP model, each layer with 500 neurons. For the learning rate, we use a decreasing scheme similar to what is suggested in [7], where after each iteration the learning rate decreases 1%. Then, each experiment's initial learning rate is tuned to achieve the best performance.

Homogeneous data distribution. The `FedCOM` algorithm is best suited for the homogeneous case, while in the heterogeneous setting it suffers from a resid-



(a) Homogeneous

(b) Heterogeneous

Figure 2: Comparing Algorithm 1 and Algorithm 2 for homogeneous and heterogeneous data distributions of the `MNIST` dataset. In heterogeneous distribution, `FedCOM` suffers from a residual error.

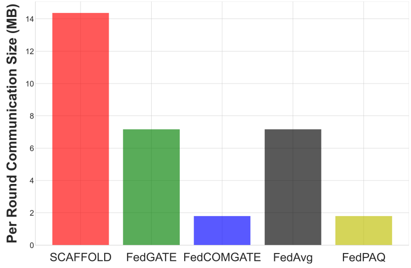


Figure 3: Communication cost at each round for the `CIFAR10` dataset with a 2-layer MLP.

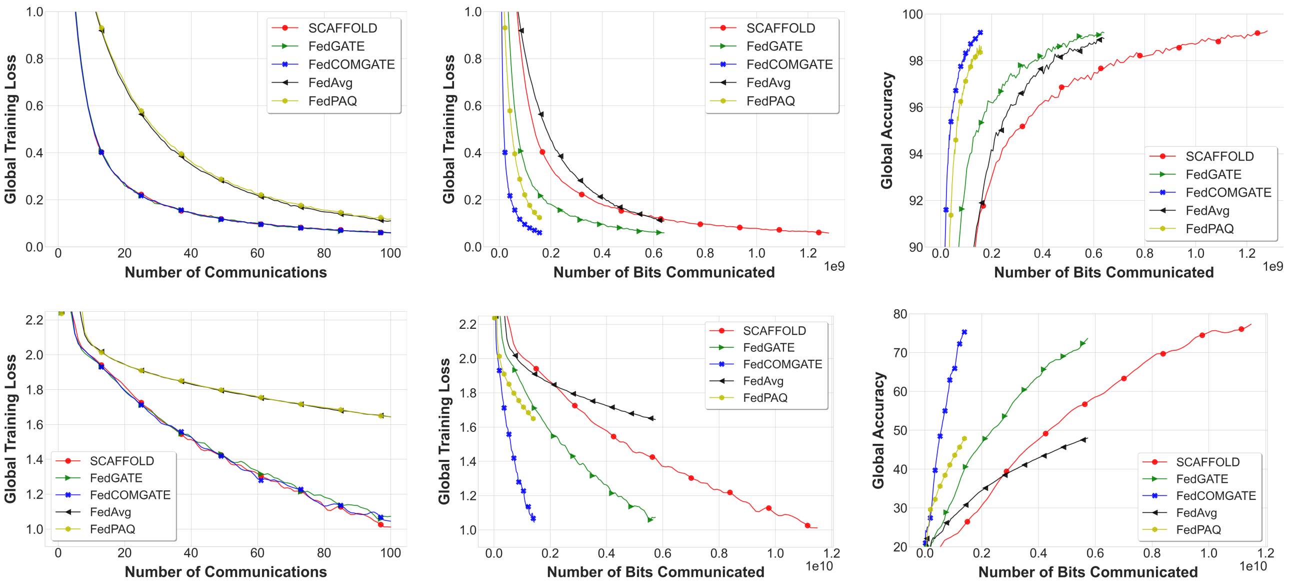
ual error. That is why we use gradient tracking in the `FedCOMGATE` algorithm. This error can be seen in Figure 2 for `MNIST` data, where in Figure 2(a) data is distributed homogeneously among devices, and in Figure 2(b) each device has access to only 2 classes in the dataset. The results indicate that we need gradient tracking in `FedCOMGATE` to deal with heterogeneity.

Heterogeneous data distribution. To generate heterogeneous data that resembles a real federated learning setup, we will follow a similar approach as in [43]. In this regard, we will distribute the data among clients in a way that each client only has data from two classes, which is highly heterogeneous. The idea behind `FedCOMGATE` is similar to the one in SCAFFOLD, except in `FedCOMGATE` we only have one control variable that gets updated using normal updates in FedAvg. In contrast, SCAFFOLD has two control variables and requires to update the global model and server control variable at each round. Hence, each client in SCAFFOLD communicates at least twice the size as `FedCOMGATE` with the server at each round, when we do not use any compression. With compression, say 4× quantization, we can substantially reduce the communication cost, say 8×, with respect to SCAFFOLD, while preserving the same convergence rate. To compare their communication cost, in Figure 3 we show the size of variables each client in each algorithm communicates with the server (for the uplink only, since the broadcasting or downlink time is negligible compared to the gathering) for the `CIFAR10` dataset with an MLP model that has 2 hidden layers, each with 500 neurons. For `FedCOMGATE` and FedPAQ we quantize the updates from 32 bits floating-point to 8 bits integer.

To show the effect of this communication size on the real-time convergence of each algorithm, we run each of them on the `MNIST` and the `CIFAR10` datasets with MLP models as described before. The data is distributed heterogeneously among clients, where each one has access to only 2 classes. Figure 4(a) shows the global model loss on training data on each communication round. FedAvg and FedPAQ are very close to each other on eacc, whereas `FedCOMGATE`, its normal version without compression `FedGATE`, and SCAFFOLD are performing similarly based on communication rounds. Figure 4(b) shows this loss based on the average number of bits communicated between each client and the server during the uplink. Also, Figure 4(c) shows the test accuracy based on this number of communicated bits. Both figures clearly demonstrate the effectiveness of proposed algorithms. Especially, the `FedCOMGATE` algorithm superbly outperforms other algorithms where the model size is relatively large.

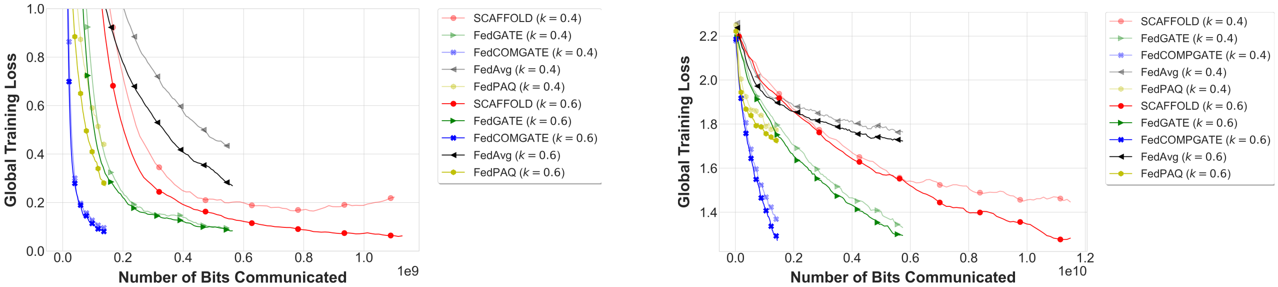
Client sampling In this section we assume that only $k \in (0, 1]$ portion of the users in the networks are active and exchange information with the server at each round. Indeed, a lower value of k implies that less nodes are active at each round and therefore the communication overhead is lower. However, it could possibly lead to a slower convergence rate and extra communications rounds to achieve a specific accuracy. We formally study the effect of k on the convergence of `FedCOMGATE` and its version without compression `FedGATE` and compare their performance with other federated methods with and without compression in Figure 5. As it can be inferred, when we decrease the k or the participation rate, generally, the performance of the model degrades with the same number of communication rounds. However, the amount of degradation might vary among different algorithms. As it is depicted in Figure 5, the proposed `FedCOMGATE` algorithm and its unquantized version, `FedGATE`, are quite robust against decreasing the participation rate between clients with respect to other algorithms such as SCAFFOLD and FedPAQ.

An important difference between algorithms proposed in this work (such as `FedCOMGATE`) and SCAFFOLD is in their performance when not all clients are active at each round of communication. Since in SCAFFOLD gradient tracking needs to be performed both locally



(a) (b) (c)

Figure 4: Comparing FedCOMGATE and FedGATE with FedAvg [43], FedPAQ [48], and SCAFFOLD [26] on the MNIST (first row) and the CIFAR10 (second row) datasets. Both FedCOMGATE and FedGATE outperform other algorithms in terms of communication size between clients and the server and convergence rate.



(a) MNIST Dataset (b) CIFAR10 Dataset

Figure 5: Comparing the effect of sampling on different algorithms. We use two datasets: the MNIST and the CIFAR10 datasets. We use an MLP with 2 layers, with 200 neurons per layer for the MNIST, and 500 neurons per layer for the CIFAR10. FedGATE and FedCOMGATE seem to be more robust against client sampling.

and globally at each round of communication, its performance could highly depend on the availability of control variate in all devices. This can lead to poor performance for SCAFFOLD when the rate of participation of clients at each round is low (as illustrated in Figure 5). This could be due to stale local control variate and new global control variate that degrade the performance of the algorithm. On the other hand, in our proposed algorithms, the gradient tracking parameter is only performed locally, and hence, the drop in the performance is much smaller than SCAFFOLD. Therefore, we think this property makes the proposed algorithms suitable for cross-device scenarios as well as cross-silo ones, whereas SCAFFOLD is more suitable for cross-silo scenarios, and not cross-device ones.

7 Conclusion

In this paper we introduced a set of algorithms for federated learning which lower the communication overhead by periodic averaging and exchanging compressed signals. We considered two separate settings: (i) homogeneous setting in which all the probability distributions and loss functions are identical; and (ii) heterogeneous setting wherein the users' distributions and loss functions could be different. For both cases, we showed that our proposed methods both theoretically and numerically require less communication rounds between server and users compared to state-of-the-art federated algorithms that use compression.

Acknowledgment

The authors would like to thank Amirhossein Reisizadeh for his comments on the first draft of the paper. We also gratefully acknowledge the generous support of NVIDIA for providing GPUs for our research. This work has been done using the Extreme Science and Engineering Discovery Environment (XSEDE) resources, which is supported by National Science Foundation under grant number ASC200045.

References

[1] MNIST dataset. http://yann.lecun.com/exdb/mnist/.

[2] A. F. Aji and K. Heafield. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 440–445, 2017.

[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] 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*, pages 5973–5983, 2018.

[5] D. Basu, D. Data, C. Karakus, and S. Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification and local computations. In *Advances in Neural Information Processing Systems*, pages 14668–14679, 2019.

[6] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569, 2018.

[7] L. Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[8] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[9] S. Caldas, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[10] Y. Deng, M. M. Kamani, and M. Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

[11] Y. Deng, M. M. Kamani, and M. Mahdavi. Distributionally robust federated averaging. *Advances in Neural Information Processing Systems*, 33, 2020.

[12] A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

[13] R. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik. Sgd: General analysis and improved rates. In *International Conference on Machine Learning*, 2019.

[14] F. Haddadpour, M. M. Kamani, M. Mahdavi, and V. Cadambe. Local sgd with periodic averaging: Tighter analysis and adaptive synchronization. *Advances in Neural Information Processing Systems*, 2019.

[15] F. Haddadpour, M. M. Kamani, M. Mahdavi, and V. Cadambe. Trading redundancy for communication: Speeding up distributed sgd for non-convex optimization. In *ICML*, pages 2545–2554, 2019.

[16] 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.

[17] F. Haddadpour, B. Karimi, P. Li, and X. Li. Fedsketch: Communication-efficient and private federated learning via sketching. *arXiv preprint arXiv:2008.04975*, 2020.

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

[19] F. Haddadpour, Y. Yang, M. Chaudhari, V. R. Cadambe, and P. Grover. Straggler-resilient and communication-efficient distributed iterative linear solver. *arXiv preprint arXiv:1806.06140*, 2018.

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] 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.

[22] N. Ivkin, D. Rothchild, E. Ullah, I. Stoica, R. Arora, et al. Communication-efficient distributed sgd with sketching. In *Advances in Neural Information Processing Systems*, pages 13144–13154, 2019.

[23] R. Jin, Y. Huang, X. He, H. Dai, and T. Wu. Stochastic-sign sgd for federated learning with theoretical guarantees. *arXiv preprint arXiv:2002.10940*, 2020.

[24] 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.

[25] H. Karimi, J. Nutini, and M. Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016.

[26] 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.

[27] 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 2020)*, 2020.

[28] M. F. F. Khan, M. M. Kamani, M. Mahdavi, and V. Narayanan. Learning to quantize deep neural networks: A competitive-collaborative approach. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2020.

[29] A. Koloskova, T. Lin, S. U. Stich, and M. Jaggi. Decentralized deep learning with arbitrary communication compression. *arXiv preprint arXiv:1907.09356*, 2019.

[30] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. U. Stich. A unified theory of decentralized sgd with changing topology and local updates. *arXiv preprint arXiv:2003.10422*, 2020.

[31] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[32] 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.

[33] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[34] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2019.

[35] Z. Li, D. Kovalev, X. Qian, and P. Richtárik. Acceleration for compressed gradient descent in distributed and federated optimization. *arXiv preprint arXiv:2002.11364*, 2020.

[36] 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.

[37] S. Lin, G. Yang, and J. Zhang. A collaborative learning framework via federated meta-learning. *arXiv preprint arXiv:2001.03229*, 2020.

[38] T. Lin, S. U. Stich, K. K. Patel, and M. Jaggi. Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217*, 2018.

[39] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *ICLR*, 2018.

[40] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations*, 2018.

[41] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[42] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[43] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[44] D. Needell, R. Ward, and N. Srebro. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in neural information processing systems*, pages 1017–1025, 2014.

[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

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

[47] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[48] 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, 2020.

[49] A. Reisizadeh, A. Mokhtari, H. Hassani, and R. Pedarsani. Quantized decentralized consensus optimization. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 5838–5843. IEEE, 2018.

[50] A. Reisizadeh, H. Taheri, A. Mokhtari, H. Hassani, and R. Pedarsani. Robust and communication-efficient collaborative learning. In *Advances in Neural Information Processing Systems*, pages 8388–8399, 2019.

[51] B. Schölkopf and A. J. Smola. *Learning with kernels*. "The" MIT Press, 2002.

[52] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[53] N. Singh, D. Data, J. George, and S. Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. *arXiv preprint arXiv:2005.07041*, 2020.

[54] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

[55] S. U. Stich. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

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

[57] 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.

[58] A. T. Suresh, F. X. Yu, S. Kumar, and H. B. McMahan. Distributed mean estimation with limited communication. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3329–3337. JMLR. org, 2017.

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

[60] 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.

[61] J. Wang, A. K. Sahu, Z. Yang, G. Joshi, and S. Kar. Matcha: Speeding up decentralized sgd via matching decomposition sampling. *arXiv preprint arXiv:1905.09435*, 2019.

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

[63] 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*, pages 1509–1519, 2017.

[64] B. Woodworth, K. K. Patel, S. U. Stich, Z. Dai, B. Bullins, H. B. McMahan, O. Shamir, and N. Srebro. Is local sgd better than minibatch sgd? *arXiv preprint arXiv:2002.07839*, 2020.

[65] J. Wu, W. Huang, J. Huang, and T. Zhang. Error compensated quantized sgd and its applications to large-scale distributed optimization. In *International Conference on Machine Learning*, pages 5325–5333, 2018.

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

[67] H. Yu, S. Yang, and S. Zhu. Parallel restarted sgd for non-convex optimization with faster convergence and less communication. *arXiv preprint arXiv:1807.06629*, 2018.

[68] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5693–5700, 2019.

[69] J. Zhang, C. De Sa, I. Mitliagkas, and C. Ré. Parallel sgd: When does averaging help? *arXiv preprint arXiv:1606.07365*, 2016.

[70] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*, 2020.

[71] 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 27th International Joint Conference on Artificial Intelligence*, pages 3219–3227, 2018.