Abstract

We propose FedGLOMO, a novel federated learning (FL) algorithm with an iteration complexity of $O(\epsilon^{-1.5})$ to converge to an $\epsilon$-stationary point (i.e., $E[\|\nabla f(x)\|^2] \leq \epsilon$) for smooth non-convex functions – under arbitrary client heterogeneity and compressed communication – compared to the $O(\epsilon^{-2})$ complexity of most prior works. Our key algorithmic idea that enables achieving this improved complexity is based on the observation that the convergence in FL is hampered by two sources of high variance: (i) the global server aggregation step with multiple local updates, exacerbated by client heterogeneity, and (ii) the noise of the local client-level stochastic gradients. The first issue is particularly detrimental to FL algorithms that perform plain averaging at the server. By modeling the server aggregation step as a generalized gradient-type update, we propose a variance-reducing momentum-based global update at the server, which when applied in conjunction with variance-reduced local updates at the clients, enables FedGLOMO to enjoy an improved convergence rate. Our experiments illustrate the intrinsic variance reduction effect of FedGLOMO, which implicitly suppresses client-drift in heterogeneous data distribution settings and promotes communication efficiency.

1 INTRODUCTION

Federated learning (FL) is a new edge-computing approach that advocates training statistical models directly on remote devices by leveraging enhanced local resources on each device (McMahan et al. [2017]). In a standard FL setting, there are $n$ clients, each having its own training data, and a central server that is trying to train a model, parameterized by $w \in \mathbb{R}^d$, using the clients’ data. Suppose the data distribution of the $i^{th}$ client is $D_i$. Then the $i^{th}$ client has an objective function $f_i(w)$ which is the expected loss, with respect to some loss function $\ell$, over data drawn from $D_i$, and the goal of the central server is to optimize the average loss $f(w)$, over the $n$ clients, i.e.,

$$f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \quad \text{and} \quad f_i(w) = E_{x \sim D_i}[\ell(x, w)].$$

The setting where the data distributions of all the clients are identical, i.e., $D_1 = \ldots = D_n$, is typically known as the “homogeneous” setting. Otherwise, the settings where the data distributions are not identical are referred to as the “heterogeneous” settings.

The core algorithmic idea of FL – in the form of FedAvg – was introduced in McMahan et al. [2017]. In FedAvg (summarized in Algorithm 3), a subset of the clients perform multiple steps of gradient descent based updates on their local data and then communicate back their respective updates to the server, which then averages them to update the global model (hence the name FedAvg). This idea of performing multiple local updates before averaging once reduces the communication cost required for training. Another essential strategy in FL is to cut down the communication cost to have the clients send compressed/quantized messages to the server in every round – this is of particular significance for training deep learning models where the number of model parameters is in millions or more.

In practice however, performing multiple local updates on clients with heterogeneous data distributions leads to the so-called phenomenon of “client drift”, wherein the individual client updates do not align well (due to over-fitting on the local client data) inhibiting the convergence of FedAvg to the optimum of the average loss over all the clients. In this paper, we identify the high variance associated with the

---

1Equal Contribution
simple averaging step of FedAvg for the global update to be at the heart of this issue.

Ever since the development of FL, significant attention has been devoted to analyzing FedAvg under different settings, modifying FedAvg using ideas from centralized optimization to accelerate the training or to reduce the communication cost; we discuss these works in Section 2. Compared to centralized optimization, a formidable challenge in the theoretical analysis of FL algorithms is the use of multiple local updates in the clients which is compounded by the heterogeneous nature of data distribution among the clients. To limit the extent of client heterogeneity, a standard assumption in FL theory is the bounded client dissimilarity (BCD) assumption, i.e.,

\[ \frac{1}{n} \sum_{i=1}^{n} \| \nabla f_i(w) - \nabla f(w) \|^2 \leq G^2 \forall w, \tag{2} \]

for some large enough constant \( G < \infty \) (e.g., see A1 in Karimireddy et al. [2020]). But this assumption is limiting as it does not allow for arbitrarily large client heterogeneity.

Recently, Arjevani et al. [2019] showed that the stochastic first-order complexity of any algorithm in the centralized setting to reach an \( \epsilon \)-stationary point (i.e., \( \mathbb{E}[\| \nabla f(x) \|^2] \leq \epsilon \)) for smooth non-convex functions is \( \Omega(\epsilon^{-1.5}) \). It is well known that vanilla SGD has a suboptimal complexity of \( O(\epsilon^{-2}) \) as it cannot mitigate the high variance of the stochastic gradient noise. Recognizing this issue, variance-reducing techniques for SGD (Fang et al. [2018], Zhou et al. [2018], Cutkosky and Orabona [2019], Liu et al. [2020]) have been proposed that attain the optimal complexity of \( O(\epsilon^{-1.5}) \).

Coming to the federated setting, as we discuss in this paper, in addition to the noise in the local client-level stochastic gradients, one has to also contend with the high variance associated with the global server aggregation step which depends on the client heterogeneity and the number of local update steps. In this case, as we argue in the subsequent sections, applying only local client-level variance-reduction is not enough for improving the iteration complexity of vanilla FedAvg beyond \( O(\epsilon^{-2}) \) for smooth, non-convex losses.

To alleviate the issue of variance due to heterogeneity, we propose a novel FL algorithm called FedGLOMO (Algorithm 1 and 2) which applies Global as well as Local variance-reducing Momentum to the server update and client updates, respectively. We prove that the iteration complexity of FedGLOMO is \( O(\epsilon^{-1.5}) \) in the smooth non-convex case, which is better than the \( O(\epsilon^{-2}) \) complexity of related works in the FL setting; see Table 1 and Theorem 1. Further, our theory does not use the BCD assumption, i.e. eq. (2), which is a standard assumption in related works. Instead, we propose and use Assumption 4 which is a more realistic and empirically verified assumption on the client drift, even allowing for arbitrary client heterogeneity. It is worth mentioning here that for FL, Karimireddy et al. [2020] also propose an algorithm (MimeMVR) which is shown to attain this improved complexity of \( O(\epsilon^{-1.5}) \) but with the BCD assumption and no compressed communication; we talk about this at the end of Section 2.

We summarize our contributions next:

(a) We propose FedGLOMO (Alg. 1 and 2), in which we apply a novel global momentum term at the server in addition to local momentum at the clients. The design of FedGLOMO is motivated by two critical issues that need to be alleviated to accelerate convergence in FL: these are the high variances associated with: (i) the global server aggregation step due to heterogeneity of clients when there are multiple local updates, and (ii) the noise of local client-level stochastic gradients. Global and local momentum result in variance reduction for the global server update and the local client updates, allowing us to tackle (i) and (ii), respectively. This enables FedGLOMO to converge to an \( \epsilon \)-stationary point (i.e., \( \mathbb{E}[\| \nabla f(x) \|^2] \leq \epsilon \)) for smooth non-convex functions in \( O(\epsilon^{-1.5}) \) gradient-based updates, which is better than the \( O(\epsilon^{-2}) \) complexity of most related works in the FL setting; see Table 1 and Theorem 1.

(b) Unlike prior work, our theory does not use the limiting bounded client dissimilarity assumption (i.e., eq. (2)). Instead, to tighten our result, we propose and use Assumption 4 – which is a novel assumption on the client drift, even allowing for arbitrary client heterogeneity in the worst case. We empirically verify that Assumption 4 holds for FedGLOMO as well as FedAvg. Theoretically, we also show that Assumption 4 holds for any FL algorithm in the case of linear regression and also with networks whose training dynamics follow that of a linearized model (a.k.a. the “NTK” regime). Refer to the discussion after Assumption 4 and Remark 2 for details.

(c) FedGLOMO is the first FL algorithm achieving \( O(\epsilon^{-1.5}) \) complexity while allowing compressed client-to-server communication. We emphasize that from the theory perspective, applying compression in FedGLOMO is not trivial and the most obvious approach does not work; see Remark 3.

(d) In Section 6, experiments on CIFAR-10 and Fashion-MNIST (Xiao et al. [2017]) show that in a highly heterogeneous setting of at most two (out of ten) classes per client, FedGLOMO requires only about one-third the number of bits used by FedAvg with PyTorch’s default momentum applied to the local client updates; see Figure 1. Our experiments also illustrate the variance reduction provided by our scheme which implicitly mitigates client-drift under heterogeneous data distribution and in turn promotes communication-efficiency.
Yu et al. [2018], Wang and Joshi [2018], Basu et al. [2019], Wang et al. [2019], Huo et al. [2020] present momentum-quantization operator of Alistarh et al. [2017]. That of Gorbunov et al. [2021]. In this work, we employ the

In Appendix D, we compare our work's complexity against however, there are no multiple local updates in these works. Lin et al. [2017], Stich et al. [2018], Basu et al. [2019], Bernstein et al. [2018], Alistarh et al. [2018], Gorbunov et al. [2021] provide distributed algorithms with improved convergence rates by also applying

3 PRELIMINARIES

Recall the setting and the optimization problem that the server is trying to solve as defined in eq. (1). We assume that the clients have access to unbiased stochastic gradients
of their individual losses. We denote the stochastic gradient of $f_i$ at $w$ computed over a batch of samples $B_i$ by $\nabla f_i(w; B_i)$. Also in this paper, $K$ is the number of communication rounds, $E$ is the number of local updates per round or the period, and $T = KE$ is the total number of local updates or the (order-wise) number of gradient-based updates. Further, $r$ is the number of clients that the server accesses in each round, i.e., the global batch size.

Vectors and matrices are written in boldface. For any positive integer $m$, the set $\{1, \ldots, m\}$ is denoted by $[m]$, and the uniform distribution over the set $\{1, \ldots, m\}$ is denoted by $\text{Unif}[0, m]$. $\mathbb{I}(.)$ is the indicator function. Next, we recap smooth functions.

**Definition 3.1 (Smoothness).** A function $g : \Theta \rightarrow \mathbb{R}$ is to be said to be $L$-smooth if for all $\theta, \theta' \in \Theta$, $\|\nabla g(\theta') - \nabla g(\theta)\| \leq L \|\theta - \theta'\|$. For all $\theta, \theta' \in \Theta$, we also have: $g(\theta') \leq g(\theta) + \langle \nabla g(\theta), \theta' - \theta \rangle + \frac{L}{2}\|\theta' - \theta\|^2$.

## 4 FEDGLOMO: GLOBAL AND LOCAL MOMENTUM-BASED VARIANCE REDUCTION

There are two issues that need to be alleviated for improving the convergence rate in FL: (i) the high variance of simple averaging used in the global server aggregation step of FedAvg, when there are multiple local updates, which is exacerbated by heterogeneity of the clients, and (ii) the high variance associated with the noise of local client-level stochastic gradients. The key idea of FEDGLOMO (Algorithm 1 and 2) is to apply variance-reducing global and local momentum to combat (i) and (ii), respectively. We now describe global and local momentum in detail.

**Global** momentum is applied to the sever aggregation step which is line 10 in Algorithm 1. To understand it better, let us revisit FedAvg (summarized in Algorithm 3) although in a slightly different way than usual) and its server aggregation step (line 12) which is just simple averaging. Similar to the update of SGD suffering from high variance, this naive averaging step – which we think of as the average of a batch of generalized stochastic gradients – is characterized by high variance stemming from heterogeneity and multiple local updates. So, this way of server aggregation slows down the convergence rate of FedAvg (and other related methods).

In this paper, we re-envision the server aggregation as a generalized gradient-based update by thinking of $(w_k - w_{k,E})$ as the generalized gradient. Then, we wish to incorporate the style of variance-reducing momentum applied in STORM (Cutkosky and Orabona 2019; Liu et al. 2020) to our gen-

### Table 1: Number of gradient updates, i.e., $T$, required to achieve $\mathbb{E}[\|\nabla f(w)\|^2] \leq \epsilon$ on smooth non-convex functions. Here, $n$ is the total number of clients and $r$ is the number of clients participating in each round. “Client Participation” asks whether all ($r = n$) or only a subset ($r < n$) of the clients participate in each round. “BCD?” asks if the bounded client dissimilarity assumption (eq. (2)) is used or not. “Compression?” asks whether compressed communication is involved or not.

<table>
<thead>
<tr>
<th>Ref.</th>
<th>$T$</th>
<th>Client Participation</th>
<th>BCD?</th>
<th>Compression?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Koloskova et al. [2020]</td>
<td>$O\left(\frac{1}{nr}\right)$</td>
<td>Full ($r = n$)</td>
<td>Yes</td>
<td>X</td>
</tr>
<tr>
<td>Haddadpour et al. [2021]</td>
<td>$O\left(\frac{1}{n^r}\right)$</td>
<td>Full ($r = n$)</td>
<td>Yes</td>
<td>✔</td>
</tr>
<tr>
<td>Khanduri et al. [2021]</td>
<td>$O\left(\frac{1}{nr}\right)$</td>
<td>Full ($r = n$)</td>
<td>Yes</td>
<td>X</td>
</tr>
<tr>
<td>Karimireddy et al. [2019]</td>
<td>$O\left(\frac{1}{n^r}\right)$</td>
<td>Partial ($r &lt; n$)</td>
<td>Yes</td>
<td>✔</td>
</tr>
<tr>
<td>Karimireddy et al. [2020]</td>
<td>$O\left(\frac{1}{n^r}\right)$</td>
<td>Partial ($r &lt; n$)</td>
<td>Yes</td>
<td>X</td>
</tr>
<tr>
<td><strong>This work (FedGLOMO)</strong></td>
<td>$O\left(\max\left(\sqrt{\frac{T}{n}}, \frac{1}{\sqrt{nr}}\right)\right)^{\alpha}$</td>
<td>Partial ($r &lt; n$)</td>
<td>No</td>
<td>✔</td>
</tr>
</tbody>
</table>

**Algorithm 1 FedGLOMO - Server Update**

1: **Input:** Initial point $w_0$, # of rounds of communication $K$, period $E$, learning rates $\{\eta_k\}_{k=0}^{K-1}$ and global batch size $r$. $Q_D$ is the quantization operator. Set $w_{-1} = w_0$.

2: for $k = 0, \ldots, K - 1$ do
3:    Server sends $w_k, w_{k-1}$ to a set $S_k$ of $r$ clients chosen uniformly at random w/o replacement.
4: for client $i \in S_k$ do
5:    Set $w_{i,0} = w_k$ and $\tilde{w}_{k-1,i} = w_{k-1}$. Run Algorithm 2 for client $i$.
6: end for
7: if $k = 0$ then
8:    Set $u_k = \frac{1}{r} \sum_{i \in S_k} Q_D(w_k - w_{i,E})$.
9: else
10:   Set $u_k = \frac{1 - \beta_k}{r} \sum_{i \in S_k} Q_D(w_k - w_{k,E}) + (1 - \beta_k) u_{k-1} + \frac{1}{r} \sum_{i \in S_k} Q_D((w_k - w_{i,E}) - (w_{k-1} - \tilde{w}_{k-1,E}))$. // (Global Momentum)
11: end if
12: Update $w_{k+1} = w_k - u_k$.
13: end for
STORM uses global server aggregation in FL. However, this equivalence produces an unbiased estimate of the input. Then defining the compression operator under Assumption 3, the compression operator is applied on $\tilde{w}_{k-1,\tau}$ over $B_{k,\tau}$, viz. $\nabla f_i(w_{k-1,\tau};B_{i,\tau}) + \nabla f_i(\tilde{w}_{k-1,\tau};B_{i,\tau})$.

Update: $v_{k,\tau} = \nabla f_i(w_{k,\tau};B_{i,\tau}) + (v_{k,\tau-1} = \nabla f_i(\tilde{w}_{k-1,\tau};B_{i,\tau})$ and $\tilde{v}_{k-1,\tau} = \nabla f_i(\tilde{w}_{k-1,\tau};B_{i,\tau}) + \tilde{v}_{k-1,\tau-1} = \nabla f_i(\tilde{w}_{k-1,\tau-1};B_{i,\tau})$.

Update $w_{k+1} = w_k - \eta_k v_{k,\tau}$. 

Send $Q_D(w_k - w_{k,E})$ and $Q_D((w_k - w_{k,E}) - (w_{k-1} - \tilde{w}_{k-1,\tau}))$ to the server.

**Algorithm 3 FedAvg [McMahan et al. [2017]]**

1: **Input:** Initial point $w_0$, # of communication rounds $K$, period $E$, learning rates $\{\eta_k\}_{k=0}^{K-1}$ and global batch size $r$.
2: for $k = 0, \ldots, K-1$ do
3: Server sends $w_k$ to a set $S_k$ of $r$ clients chosen uniformly at random w/o replacement.
4: for client $i \in S_k$ do
5: Set $w_{k,i} = w_k$.
6: for $\tau = 0, \ldots, E - 1$ do
7: Pick a random batch of samples in client $i$, $B_{i,\tau}$. Compute the stochastic gradient of $f_i$ at $w_{k,i}$ over $B_{i,\tau}$, viz. $\nabla f_i(w_{k,i};B_{i,\tau})$.
8: Update $w_{k,i+1} = w_{k,i} - \eta_k \nabla f_i(w_{k,i};B_{i,\tau})$.
9: end for
10: Send $(w_k - w_{k,E})$ to the server.
11: end for
12: Update $w_{k+1} = w_k - \frac{1}{r} \sum_{i \in S_k} (w_k - w_{k,E})$.
13: end for

Now that we understand global momentum, let us move on to local momentum. For this see lines 3, 6 and 8 in Algorithm 2; these give us $(w_k - w_{k,E})$ and $(w_{k-1} - \tilde{w}_{k-1,E})$ after running for $E$ steps. But notice that these lines are the same as eq. (3) with $\beta_j = 0$ and the stochastic gradient at the first iteration replaced by the full gradient. It is worth mentioning here that these local updates are also similar to SPIDER which is an SVRG-style update proposed in [Fang et al. 2018]. However, recognizing that this is also a special case of the STORM update with $\beta_j = 0$, we prefer calling it momentum in order to have a unifying terminology for both the global and local updates.

One might wonder what is the role of global momentum as SPIDER can be extended to improve the complexity in distributed optimization without multiple local updates. For this, Appendix F, we consider FedGLOMO (Algorithm 4 and 5 in the Appendix) which is a simpler version of FedGLOMO with only local momentum and no global momentum (i.e., plain averaging at the server which is equivalent to setting $\beta_k = 1$ in Algorithm 1), and show that it does not achieve $O(\epsilon^{-1.5})$ complexity under partial-device participation and compression (see Theorem 3 in Appendix). The root cause of this is client heterogeneity which amplifies its effect under multiple local updates; without incorporating some form of variance reduction in the server aggregation step, the complexity cannot be improved.

Let us try to provide some intuition as to how incorporating global momentum helps. Suppose we keep $\eta_k = \eta$ and $\beta_k = \beta < 1$ for all $k$. Theoretically, we get a lower complexity.
bound for $\beta$ which is $O(\eta^2)$. Then with this momentum-based aggregation strategy, the variance reduces by a factor of $O(\beta/\eta) = O(\eta)$ as compared to aggregation by plain averaging. (There are some other terms too but these are sufficiently small.) This reduction in the variance by a factor of $O(\eta)$ is what improves the convergence rate of FedGLOMO.

It is true that FedGLOMO has to communicate twice the amount of information per round as compared to FedAvg or FedPAC [Reisizadeh et al. 2020] which is just FedAvg with compressed communication. One can set the precision that for any $\alpha \leq \frac{1}{m}$ (or any stochasticity in the local updates. Let $\tau$, $\sigma$, and $\beta$ be the number of local steps and $\alpha$ of $A$ at the start of the $t$th communication rounds. Let $w^*$ $\in \mathbb{R}^D$ be the client's local parameter $\sum_{i \in [n]} x_i$, so $\alpha \leq 0$ provably for any FL algorithm. Suppose in client $i$, we have feature and label pairs $(x, y) \sim (\mathcal{X}_i, \mathcal{Y}_i)$, where the label

$$y = \langle w^*, x \rangle + \xi,$$

with $\xi \sim \mathcal{N}(0, \sigma^2)$ being independent zero-mean client-dependent random noise. Obviously, the label distribution $\mathcal{Y}_i$ here depends on the feature distribution $\mathcal{X}_i$, noise distribution $\mathcal{N}_i$ and $w^*_i$. We assume that the covariance matrix of the feature vectors is the same across all the clients, i.e., $\mathbb{E}_{x \sim \mathcal{X}_i} [xx^T] = Q$ for all $i \in [n]$: this is possible for e.g., by normalization or whitening of the features. Note that by assuming the same covariance matrix across all the clients, we are not assuming that the feature distributions are the same across clients, but even if they are, there is heterogeneity through the different label distributions. Then, with the squared loss, our per-client objective function is:

$$f_i(w) = \mathbb{E}_{(x, y) \sim (\mathcal{X}_i, \mathcal{Y}_i)} \left[ \frac{1}{2} (y - \langle w, x \rangle)^2 \right].$$

With the aforementioned conditions, it can be verified that $\nabla f_i(w) = Q (w - w^*_i)$. Thus, $\nabla f_i(w) = Q (w - w^*_i)$, and so $\sum_{i \in [n]} e_{k,\tau}^{(i)} = 0$. So, Assumption 4 holds here with $\alpha = 0$ for any FL algorithm.

In fact, the above analysis and result (i.e., $\alpha = 0$) can be extended to networks whose training dynamics follow that of a linearized model, which has been shown to be the case for infinite-width networks [see e.g., Lee et al. 2019 and Jacot et al. 2018] and has been also used on
applications for finite-width networks (for e.g., in [Mu et al., 2020]).

We now present the abridged version of the convergence result of FedGLOMO, followed by some important remarks. Its full version and detailed proof are in Appendix A and G.1, respectively.

**Theorem 1 (Smooth non-convex).** Let Assumptions [7, 2, 3] hold. Further, suppose Assumption 4 is true for FedGLOMO. In FedGLOMO, for each round $k$, set $\eta_k = \eta = \mathcal{O}(\frac{1}{LEKT})$, where $C = \mathcal{O}(\max(\frac{n}{\alpha}, E^{2((1+q)/2)}))$, and $\beta_k = \mathcal{O}((1+q)\eta^2 L^2 E^4)$. Suppose we use full-device participation (i.e., the global batch size is $n$) only at $k = 0$. Then, FedGLOMO can achieve $E_{k^*} \equiv \sum_{i \in [n]}[\|\nabla f(w_{k^*})\|^2] \leq \epsilon$ in $K = \mathcal{O}(\max(\sqrt{\frac{n}{\alpha}}, 1+\epsilon)^{1.5})$ rounds of communication and $E = \mathcal{O}(1/n)$ local steps.

**Remark 1 (Better iteration complexity).** As per Theorem 1 for converging to an $\epsilon$-stationary point, FedGLOMO needs $T = KE$ to be $\mathcal{O}(\max(\sqrt{\frac{n}{\alpha}}, 1+\epsilon)^{1.5})$. This iteration complexity is the same as that of MimeMVR (Karimireddy et al., 2020) but without using the bounded client dissimilarity assumption, i.e., eq. [2], (also see the next remark for more details on this) and better than other related works in the federated setting; see Table 1. We underscore the significance of global momentum here by comparing this complexity of FedGLOMO to that of FedLOMO (recall this is a simpler version of FedGLOMO with only local momentum and no global momentum, described in Appendix F) under partial-device participation and compression which is $\mathcal{O}(\frac{1}{\epsilon} n^{-2})$; see Theorem 3 in the Appendix.

**Remark 2 (No requirement of bounded client dissimilarity (BCD) assumption).** Divergent from related works, Theorem 1 does not use the commonly used BCD assumption, i.e., eq. [2]. This is achieved by utilizing the smoothness and non-negativity of the $f_i$’s, specifically $\frac{1}{n} \sum_{i \in [n]}[\|\nabla f_i(w)\|^2] \leq \frac{1}{n} \sum_{i \in [n]}[2L(f_i(w) - f_i) \leq 2L f(w)];$ see the proof outline of Theorem 1 in Appendix A. Instead of the BCD assumption, we use our empirically verified Assumption 4 to provide a tighter (when $\alpha \ll n$) and data-dependent convergence result. Note that Assumption 4 will always hold for some $\alpha \ll n$, regardless of the degree of client heterogeneity. Thus, Theorem 1 allows for arbitrary client heterogeneity.

**Remark 3 (Compressed communication).** To our knowledge, FedGLOMO is the first algorithm that attains the aforementioned improved iteration complexity for FL on smooth non-convex functions with compressed communication. We emphasize that the choice of quantities compressed in line 10 of Algorithm 1 is important. This particular choice enables deriving the improved complexity by first deriving a result analogous to smoothness, i.e., $\| (w_k - w_{k, E}) - (w_{k-1} - w_{k-1, E}) \| \leq \tilde{L} \| w_k - w_{k-1} \|$ (see Lemma 9 in Appendix G.1). The straightforward choice of sending $Q_D(w_k - w_{k, E})$ and $Q_D(w_{k-1} - w_{k-1, E})$ prohibits us from deriving the improved rate, unless we also assume $Q_D(.)$ to be a Lipschitz operator.

In Appendix B, for $r \ll n$, we show that using the quantization scheme of [Alistarh et al., 2017] with $s = \sqrt{d}$, FedGLOMO achieves more than a five-fold saving in the total communication cost as compared to when there is full-precision communication in FedGLOMO.

**Remark 4 (A limitation).** Even though our iteration complexity of $T = \mathcal{O}(\epsilon^{-1.5})$ is better than that of FedCOMGATE proposed by [Haddadpour et al., 2021] (which is $\mathcal{O}(\epsilon^{-2})$), our communication complexity of $K = \mathcal{O}(\epsilon^{-1.5})$ is higher than that theirs which is $K = \mathcal{O}(\epsilon^{-1})$ (albeit under an extra assumption on the quantizer, namely Assumption 5 in their paper). Ideally, we would like to have $E = \mathcal{O}(\epsilon^{-p})$ and $K = \mathcal{O}(\epsilon^{-(1.5-p)})$ for some $p > 0$, in order to reduce FedGLOMO’s communication complexity. Exploring whether such a result is obtainable with our proposed style of momentum is an interesting future direction.

6 EXPERIMENTS

To show the efficacy of global momentum in FedGLOMO, we compare it against FedLOMO (recall this has only local momentum and no global momentum; see Appendix F) and FedAvg [McMahan et al., 2017] with the standard momentum available in PyTorch applied to (i) only its local updates, and (ii) both local and global updates — all with compressed client-to-server communication. We denote (i) and (ii) by FedAvg-Im and FedAvg-glm (“Im” and “glm” stand for local momentum, and global + local momentum), respectively. FedAvg with compression is referred to as FedPAQ (Reisizadeh et al., 2020). Similarly, we call FedAvg-Im and FedAvg-glm with compression, as FedPAQ-Im and FedPAQ-glm. We also compare against FedCOMGATE [Haddadpour et al., 2021] which uses gradient tracking to theoretically derive a better communication-complexity than us (see Remark 4). For compression, the “qsgrd” operator proposed in [Alistarh et al., 2017] is used.

We consider the task of classification on CIFAR-10 and Fashion-MNIST [Xiao et al., 2017] abbreviated as FMNIST henceforth. The model used is a two-layer neural network with ReLU activation in the hidden layers. The size of both the hidden layers is 300/600 for FMNIST/CIFAR-10. We train the models using the categorical cross-entropy loss with $\ell_2$-regularization. The weight decay value in PyTorch — regularization) is set to 1e-4. We consider both homogeneous and heterogeneous data distribution among the clients. Similar to [McMahan et al., 2017] for the heterogeneous case, we distribute the data among the clients such that each client can have data from either one or (at most) two classes — note that this is a high degree of heterogeneity. The exact procedure is described in Appendix E.
ber of clients ($n$) in all the experiments is set to 50, with each client having the same number of samples. The global batch-size $r$ is 25, and the number of local updates per round (i.e., $E$) is 10. All full gradients are replaced by stochastic gradients computed on a (per-client) batch size of 256. The learning rates, momentum parameters of the algorithms, and some other experimental details are in Appendix E.

In Fig. 1 we compare FedPAQ-lm, FedPAQ-glmlm, FedGLOMO and FedCOMGATE with 4 (resp., 8) bits per-round against FedGLOMO with 2 (resp., 4) bits per-round on FMNIST (resp., CIFAR-10) in the heterogeneous and homogeneous cases. We set the number of per-round bits used by FedGLOMO to be half the number used by all other algorithms, so that each one has the same per-round communication budget. All plots depict results over 3 independent runs; the shaded regions represent $\pm 1$ standard deviation whereas the solid lines are the respective means. Please see the discussion in the figure caption. These results illustrate the power of global momentum.

Next, in the no-compression heterogeneous case, we compare against Mime (specifically, “MimeSGDm”) of Karimireddy et al. [2020] which also attains a complexity of $O(\epsilon^{-1.5})$ but without compressed communication, and is tailored to handle client heterogeneity. Having shown the suboptimality of FedGLOMO and FedPAQ-lm in Fig. 1, we only compare FedAvg-glmlm, FedGLOMO without compression and MimeSGDm in the heterogeneous case in Fig. 2. The plots in Fig. 2 show that the implicit client-drift controlling ability of our proposed global momentum is on par with the explicit client-drift controlling mechanism of Mime. The test error values averaged over the last five rounds for the plots in Figures 1 and 2 are in Tables 2 and 3 respectively.

We also provide some more empirical results on CIFAR-100 in Appendix E.1.

![Table 2: Average test error % (± standard deviation) over the last five rounds for the plots in the heterogeneous (top) and homogeneous (bottom) cases in Figure 1.](attachment:image2.png)

<table>
<thead>
<tr>
<th>Algo.</th>
<th>CIFAR-10 Het.</th>
<th>FMNIST Het.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FedPAQ-lm</td>
<td>50.26 ± 0.85</td>
<td>16.17 ± 0.53</td>
</tr>
<tr>
<td>FedPAQ-glmlm</td>
<td>49.88 ± 1.15</td>
<td>15.87 ± 1.10</td>
</tr>
<tr>
<td>FedGLOMO</td>
<td>53.74 ± 0.17</td>
<td>18.95 ± 0.19</td>
</tr>
<tr>
<td>FedCOMGATE</td>
<td>46.26 ± 0.05</td>
<td>13.55 ± 0.32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algo.</th>
<th>CIFAR-10 Hom.</th>
<th>FMNIST Hom.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FedPAQ-lm</td>
<td>45.13 ± 0.07</td>
<td>13.08 ± 0.05</td>
</tr>
<tr>
<td>FedPAQ-glmlm</td>
<td>45.70 ± 0.10</td>
<td>11.76 ± 0.06</td>
</tr>
<tr>
<td>FedGLOMO</td>
<td>45.96 ± 0.01</td>
<td>14.22 ± 0.01</td>
</tr>
<tr>
<td>FedCOMGATE</td>
<td>44.97 ± 0.05</td>
<td>10.98 ± 0.05</td>
</tr>
</tbody>
</table>

Table 3: Average test error % (± standard deviation) over the last five rounds for the plots in Figure 2.

Verifying Assumption 3 for FedGLOMO: For each round $k$, we compute $\alpha = \max_{\tau \in [E]} \frac{\sum_{i \in [n]} \| \tilde{e}_{k,\tau}^{(i)} \|^2}{\sum_{i \in [n]} \| e_{k,\tau}^{(i)} \|^2}$, where $e_{k,\tau}^{(i)}$ is as defined in Assumption 3 for 4 and 2 bit FedGLOMO on CIFAR-10 and FMNIST, respectively. Note that we remove the expectation (w.r.t. the stochastic gradients) while computing $\alpha$ for empirical verification. In Fig. 2 we plot $(\alpha/n)$ over different rounds for the heterogeneous as well as homogeneous case on both datasets; see the discussion in the figure caption.

7 CONCLUSION

We presented FedGLOMO, a communication-efficient algorithm for faster federated learning via the application of variance-reducing momentum, both in the aggregation step at the server as well as local client updates. We showed that FedGLOMO has better iteration complexity than prior work on smooth non-convex functions with compressed communication. Further, unlike prior work, our result does not use the bounded client dissimilarity assumption, even holding under arbitrary client heterogeneity. We also demonstrate the efficacy of FedGLOMO via extensive experiments.

Acknowledgements

This work is supported by NSF grants CCF-1564000, IIS-1546452 and HDR-1934932, AFOSR grant FA9550-19-1-0005, and NASA grant 80NSSC21M0071.

References


Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds
Figure 1: Comparison of FedPAQ-lm, FedPAQ-glm, FedLOMO, FedGLOMO and FedCOMGATE (Haddadpour et al., 2021) with the same per-round communication budget on FMNIST and CIFAR-10 in the heterogeneous (top four figs.) and homogeneous (bottom four figs.) settings, respectively. The x-axis is the total number of communicated bits divided by the dimension $d$ and the global batch-size $r$. FedGLOMO is the fastest and most communication-efficient algorithm in almost all the cases; for e.g., in the heterogeneous case for both datasets, FedGLOMO attains the final test error of FedPAQ-glm (resp., FedPAQ-lm) with less than a half (resp., only about a third) of the number of bits used by FedPAQ-glm (resp., FedPAQ-lm). Further, FedGLOMO and FedLOMO have a smoother trajectory than other algorithms in the heterogeneous case due to variance-reducing momentum. Observe that FedLOMO and FedPAQ-lm (with only local momentum) are slower than FedGLOMO and FedPAQ-slm (with both local and global momentum), showing the ineffectiveness of only local momentum and the power of combining both local and global momentum. Also, note that FedGLOMO performs much better than FedCOMGATE in the homogeneous case.

Figure 2: Comparison of FedAvg-glm, FedGLOMO (without compression) and MimeSGDm on FMNIST and CIFAR-10 in the heterogeneous case. On both datasets, FedAvg-glm is the slowest while FedGLOMO is somewhat faster than MimeSGDm. While Mime has an explicit client-drift control mechanism, we do not have that in FedGLOMO, but still our proposed global momentum implicitly mitigates client-drift as well as Mime.

Figure 3: Variation of $\frac{\alpha}{n}$ over different rounds of 4 and 2 bit FedGLOMO for CIFAR-10 (Fig. 3a) and FMNIST (Fig. 3b) in the heterogeneous and homogeneous cases. In both cases, notice that $\alpha \ll n$ throughout training. Also, as discussed after the statement of Assumption 4 note that $\frac{\alpha}{n}$ is higher for the heterogeneous case (except at the end of training for FMNIST). See Figure 4 in the Appendix for the same on FedAvg.


Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-sgd: Distributed sgd with quanti-


