On the Convergence of Hybrid Federated Learning with Server-Clients Collaborative Training

Kun Yang, Cong Shen
Department of Electrical and Computer Engineering
University of Virginia
Charlottesville, VA 22904
{ky9tc,cong}@virginia.edu

2022 56th Annual Conference on Information Sciences and Systems (CISS) | 978-1-6654-1796-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CISS53076.2022.9751161

***Abstract*—State-of-the-art federated learning (FL) paradigms utilize data collected and stored in massively distributed clients to train a global machine learning (ML) model, in which local datasets never leave the devices and the server performs simple model aggregation for better privacy protection. In reality, however, the parameter server often has access to certain (possibly small) amount of data, and it is computationally more powerful than the clients. This work focuses on analyzing the convergence behavior of hybrid federated learning that leverages the server dataset and its computation power for collaborative model training. Different from standard FL where stochastic gradient descent (SGD) is always computed in a parallel fashion across all clients, this architecture enjoys both parallel SGD at clients and sequential SGD at the server, by using the aggregated model from clients as a new starting point for server SGD. The main contribution of this work is the convergence rate upper bounds of this aggregate-then-advance hybrid FL design. In particular, when the local SGD keeps an $\mathcal{O}(1/t)$ stepsize, the server SGD must adjust its stepsize to scale no slower than $\mathcal{O}(1/t^2)$ to strictly outperform local SGD with strongly convex loss functions. Numerical experiments are carried out using standard FL tasks, where the accuracy and convergence rate advantages over clients-only (FEDAVG) and server-only training are demonstrated.**

I. INTRODUCTION

Modern machine learning (ML) paradigms largely reside at two extremes. The first category is the *centralized* model training, where the training/validation (and even testing) data are completely accessible by a server, which is also equipped with sufficient computation resources to carry out complex deep neural network (DNN) model training tasks. This centralized learning paradigm has been the *de facto* architecture in deep learning, but also has two major shortcomings. First, overfitting often happens when the server does not possess large amount of data. In this case, the limited data do not provide sufficiently accurate statistical information, which leads to many popular optimization methods such as stochastic gradient descent (SGD) to converge to suboptimal models. Second, the sequential nature of SGD may significantly slow down the convergence time, especially compared with the linear speedup distributed SGD enjoys [1].

To address the issues of centralized training, the second category of *distributed* model training has received a lot of interest over the past years [2]. In fact, real-world large-scale problems often rely on distributed computing architectures [3], [4], such as the server/clients paradigm in parallel SGD [1], local SGD [5], and federated learning (FL) [6]. This architecture is the opposite of the first category: the clients have all the training data and carry out the heavy workload of computing stochastic gradients over local data, while the server simply aggregates the updated model periodically. The distributed paradigm has several attractive properties such as the (often linear) speedup in model convergence [5], [7], [8], better data privacy [6], and easy scalability to adding new clients and new local data [9]. The downside, however, is that the server is relegated to performing very simple calculations, which is a significant waste of its computation resources.

This paper focuses on analyzing the convergence behavior of a hybrid federated learning paradigm, which assumes that the parameter server also has some amount of data that can be potentially used for ML model training. In particular, we focus on the convergence analysis of a simple aggregate-then-advance hybrid design, which we call *cascading local-global SGD (CLG-SGD)*. This paradigm sequentially concatenates the standard local SGD with an episode of server SGD that starts with the latest averaged global model and advances the model using the server dataset. This simple and yet intuitive design allows the server to utilize its strong computational power and available (possibly limited) dataset to improve the global model. However, its convergence analysis is substantially more difficult due to the *heterogeneous mixture* of local SGD and server-only SGD.

The main contribution of this paper is the convergence analysis of CLG-SGD that highlights its convergence advantage over local SGD. The analysis establishes that when the local SGD keeps an $\mathcal{O}(1/t)$ stepsize, the server SGD must adjust its stepsize to scale no slower than $\mathcal{O}(1/t^2)$ to strictly outperform local SGD with strongly convex loss functions. In addition to the theoretical analysis, we also carry out numerical experiment using standard MNIST and CIFAR-10 classification tasks to evaluate the empirical benefits of CLG-SGD over FEDAVG and server-only training.

The remainder of this paper is organized as follows. The system model and problem formulation are presented in Section II. CLG-SGD is given in Section III and analyzed

The work is partially supported by the National Science Foundation under Grant ECCS-2033671.

in Section IV. Numerical experiment results are reported in Section V. Finally, Section VI concludes the paper.

II. System Model and Problem Formulation

The optimization problem. We consider the standard empirical risk minimization problem

$$\min_{x\in\mathbb{R}^d} f(x) = \min_{x\in\mathbb{R}^d} \frac{1}{m}\sum_{z\in\mathcal{D}} l(x;z), \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is the differentiable loss function averaged over the total dataset $\mathcal{D}$ with size m, $x \in \mathbb{R}^d$ is the machine learning model variable that one would like to optimize, and $l(x;z)$ is the loss function evaluated at data sample z and model x. We assume that there are n distributed clients in the system. The problem (1) can be rewritten as

$$\min_{x\in\mathbb{R}^d} f(x) = \min_{x\in\mathbb{R}^d} \sum_{i=1}^{n} \frac{m_i}{m} f_i(x), \tag{2}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is the local loss function for client i, averaged over its local dataset $\mathcal{D}_i$ with m_i data samples, and $\sum_i m_i = m$, i.e.,

$$f_i(x) = \frac{1}{m_i}\sum_{z\in\mathcal{D}_i} l(x;z), \forall i = 1,\cdots,n.$$

For simplicity, we assume that $m_i = m_j, \forall i \neq j$ for the remainder of this paper. Furthermore, let f^* and f_i^* be the minimum values of f and f_i, respectively, i.e.,

$$x^* := \arg\min_{x\in\mathbb{R}^d} f(x), f^* := f(x^*);$$
$$x_i^* := \arg\min_{x\in\mathbb{R}^d} f_i(x), f_i^* := f_i(x_i^*), \forall i = 1,\cdots,n.$$

In this conference paper, the convergence analysis focuses on the IID dataset setting across both clients and server, and we leave the analysis of non-IID local client datasets to the journal version. The IID setting assumes all local datasets $\{\mathcal{D}_i\}$ are sampled IID from $\mathcal{D}$ with distribution ν, which is the underlying data distribution for $\mathcal{D}$.

Local SGD. We assume that each client runs an (independent) SGD sequence with E SGD steps (iterations) in every communication round. We use T to denote the total communication rounds. At the t-th round, client i receives the latest global model x_t from the parameter server, and starts E SGD steps of stochastic gradient evaluations:

$$x_{t,0}^i = x_t; x_{t,\tau+1}^i = x_{t,\tau}^i - \eta_t \nabla f_i(x_{t,\tau}^i), \tau = 0,\cdots,E-1.$$

To simplify the notation we use $f_i(x) = l(x;\xi_i)$ to denote the loss function of model x with a random data sample ξ_i at client i (or equivalently, the i-th parallel SGD sequence). After E steps of the parallel SGD, client i has the local model $x_{t+1}^i = x_{t,E}^i$ and the parameter server collects the local models $\{x_{t+1}^i\}_{i\in[n]}$ and computes a simple aggregation $x_{t+1} = \frac{1}{n}\sum_{i=1}^n x_{t+1}^i$. Local SGD then moves on to the $(t+1)$-th round.

Server-only SGD. We assume that the server has access to a "local" (i.e., only by the server) dataset $\mathcal{D}_s$ with size m_s, and the server can perform model training on this dataset. The latent distribution that generates the server dataset is the same ν so that we have a consistent (in expectation) optimization problem as (1), i.e., every data sample $z \in \mathcal{D}_s$ is drawn IID from distribution ν.

With the server dataset $\mathcal{D}_s$, the ML model can be trained at the server to solve the following problem:

$$\min_{x\in\mathbb{R}^d} f_s(x) = \min_{x\in\mathbb{R}^d} \frac{1}{m_s}\sum_{z\in\mathcal{D}_s} l(x;z). \tag{3}$$

This problem can be efficiently solved via SGD, which computes the gradient using one random data sample each time. However, when $m_s \ll m$, solving problem (3) often does not solve the original problem (1).

III. Cascading Local and Global SGD

Algorithm 1: CLG-SGD

Initialization: Server initializes x_0;
for $t = 0$ **to** $T-1$ **do**
  // Server action
  Server broadcasts x_t to all clients;
  // Clients actions in parallel
  for *client* $i \in [n]$ **do**
    $x_{t,0}^i \leftarrow x_t$;
    for $\tau = 0$ **to** $E-1$ **do**
      $x_{t,\tau+1}^i = x_{t,\tau}^i - \eta_t \nabla f_i(x_{t,\tau}^i)$
    end
    $x_{t+1}^i \leftarrow x_{t,E}^i$;
    Uploads x_{t+1}^i to the server;
  end
  // Server action after clients upload
  Server aggregates $x_{t+1}^s = \frac{1}{n}\sum_{i=1}^n x_{t+1}^i$;
  $x_{t+1,0}^s \leftarrow x_{t+1}^s$;
  for $\tau = 0$ **to** $K-1$ **do**
    $x_{t+1,\tau+1}^s = x_{t+1,\tau}^s - \gamma_t \nabla f_s(x_{t+1,\tau}^s)$
  end
  $x_{t+1} \leftarrow x_{t+1,K}^s$;
end
Output: x_T

A simple extension of local SGD [5] is presented in Algorithm 1 to utilize the server computation power and dataset. We note that this routine can be easily extended to account for other characteristics, such as partial clients participation and imbalanced/non-IID local datasets in federated learning, but we choose to keep it as is to facilitate its theoretical analysis. In Algorithm 1, the client and server gradient computations are carried out in a *cascading* fashion, with model aggregation mixed in between. In particular, after receiving the locally updated models from clients $\{x_t^i\}_{i\in[n]}$ and aggregating to have an updated global model x_{t+1}^s, the server uses this new model as a new starting point and

proceeds to advance the global model to x_{t+1} by running SGD for K steps. We consider the data center setting where both local and global SGDs can uniformly randomly sample the training dataset. However, the server may choose a stepsize that can be different from the clients. Then, the server broadcasts the aggregated-then-advanced global model x_{t+1} to clients for round $t+1$.

Intuitively, CLG-SGD should improve the performance of local SGD for the same number of communication rounds, which generally dominate the overall cost of the system [10] and thus must be carefully controlled. This intuition will be made precise in the subsequent convergence analysis. We also comment that since the server training has to wait for the aggregation of client model update to complete, CLG-SGD has greater wall-clock delay than local SGD. However, since the parameter server is often much more computationally powerful than clients, this additional model training to advance the global model from x_{t+1}^s to x_{t+1} incurs much smaller delay compared with client model training.

IV. Convergence Analysis

It is well-known that the key difficulty in analyzing local SGD-type algorithms lies in proving that they can decrease the variance of the global model across communication rounds, and the degree of variance reduction determines the convergence rate. For the analysis of CLG-SGD, however, this aspect of variance reduction must be characterized over a *heterogeneous concatenation* of both parallel and serial SGDs in one communication round. More generally, this can be viewed as analyzing variance reduction of the concatenation of a n-level local SGD (small variance) and a 1-level server SGD (large variance). Notably, the n local SGD processes do not share the same starting point as the 1-level server SGD, which is fundamentally different than the many convergence analyses on local SGD/FedAvg and their variants [1], [5], [7], [8], [11]–[13].

In this section, we present a convergence analysis of CLG-SGD. Both strongly convex and non-convex loss functions are analyzed. The analyses for both categories start with $E = 1$ (recall E is the number of iterations for local SGD) and then extend to $E > 1$. The reason to explicitly analyze $E = 1$ is that this is a rather simple configuration which allows us to focus on the key technical challenges associated with cascaded local and global SGD[1]. In addition, this simple case enables a rigorous comparison of CLG-SGD against vanilla local SGD under the same configuration, and we are able to prove that, by choosing the server stepsize γ_t carefully, CLG-SGD can strictly outperform local SGD for the same communication rounds. For both categories of loss functions, we always assume $K = 1$ to simplify the analysis, i.e., the server only runs one iteration of SGD. Extension to $K > 1$ can be done in a way similar to how we extend from $E = 1$ to $E > 1$. Due to the space limitation, we omit the technical proofs, which will be provided in the journal version.

[1]In fact, $E = 1$ can be viewed as cascading *parallel SGD* [3], [4] and server SGD.

A. Strongly Convex Loss Functions

We limit our attention to L-smooth and μ-strongly convex loss functions in this subsection, as stated in Assumptions 1 and 2. In addition, we assume that the stochastic gradients are unbiased and the variance is bounded in Assumption 3.

Assumption 1 *$l(x,\xi)$ is L-smooth: $\|\nabla l(x,\xi) - \nabla l(y,\xi)\| \leq L\|x-y\|$ for any $x, y \in \mathbb{R}^d$ and any $\xi \in \mathcal{D}$.*

Assumption 2 *$l(x,\xi)$ is μ-strongly convex: $\langle \nabla l(x,\xi) - \nabla l(y,\xi), x-y\rangle \geq \mu\|x-y\|^2$ for any $x, y \in \mathbb{R}^d$ and any $\xi \in \mathcal{D}$.*

Assumption 3 *SGD is unbiased: $\mathbb{E}\nabla l(x,\xi) = \nabla L(x)$, and its variance is bounded: $\mathbb{E}\|\nabla l(x,\xi) - \nabla L(x)\|^2 \leq \sigma^2$.*

Our main result is given in Theorem 1.

Theorem 1 *Let Assumptions 1 to 3 hold. For $E = 1$ and set the stepsizes for both local and server SGD as $\eta_t = \gamma_t = \frac{1}{\mu t}$, there exists a constant $t_0 = \frac{L^2}{\mu^2}$ such that for any $t > t_0$, the convergence of CLG-SGD satisfies*

$$\mathbb{E}\|x_t - x^*\|^2 \leq \frac{t_0}{t}\mathbb{E}\|x_{t_0} - x^*\|^2 + \frac{C_0}{t}, \tag{4}$$

where $C_0 \triangleq \frac{2\sigma^2}{\mu^2}(1 + \frac{1}{n})$.

Theoretical comparison of CLG-SGD and local SGD. A straightforward evaluation of Eqn. (4) reveals the same $\mathcal{O}(1/(nt))$ asymptotic convergence behavior, which has the same well-known linear speedup in n. The important and probably more interesting question regarding Theorem 1 is how it compares with the convergence rate of local SGD. Our derivation can recover the known convergence rate of local SGD by removing the server training component. We note, however, that setting the same stepsize of $\eta_t = \gamma_t = 1/(\mu t)$ as in Theorem 1 cannot produce a definitive comparison of the two convergence rate upper bounds. This is because although the convergence coefficient reduces in CLG-SGD, the overall SGD noise power increases due to additional SGD steps at the server. This calls for a more careful investigation into the choice of server stepsize γ_t, which leads to the following result.

Proposition 1 *Let Assumptions 1 to 3 hold. For $E = 1$ and set the clients stepsize as $\eta_t = \frac{1}{\mu t}$, the convergence rate upper bound of CLG-SGD in Theorem 1 is no larger than that of local SGD under the same communication configurations when the server stepsize satisfies*

$$\gamma_t \leq \frac{2}{(L^2+1)\, n\mu t^2} \sim \mathcal{O}\left(\frac{1}{t^2}\right). \tag{5}$$

Eqn. (5) states that the scaling of stepsize should be no slower than $\mathcal{O}(1/t^2)$. We note that it is well established that

local SGD only requires the stepsize to scale at $\mathcal{O}(1/t)$ for strongly convex loss functions [1], [5], [7], [11]. Indeed, Theorem 1 states that this is still sufficient to achieve the $\mathcal{O}(1/t)$ convergence rate for CLG-SGD. However, if we further require that CLG-SGD outperforms local SGD, the server stepsize has to be more stringently controlled to simultaneously reduce the convergence coefficient and overall SGD noise, which leads to Eqn. (5).

Building on Theorem 1, we now relax the assumption of $E = 1$ and study multiple SGD steps ($E > 1$) at the clients. The main result is presented in Theorem 2.

Theorem 2 *Assume Assumptions 1 to 3 hold and $E > 1$. If the stepsizes are set as $\eta_t = \gamma_t = \frac{4}{\mu t}$, there exists a constant $t_0 = \frac{4L^2}{\mu^2}$ such that for any $t > t_0$, the convergence of CLG-SGD with strongly convex loss functions satisfies*

$$\mathbb{E}\|x_t - x^*\|^2 \leq \frac{t_0}{t}\mathbb{E}\|x_{t_0} - x^*\|^2 + \frac{D_1}{t} + \frac{D_2}{t^2} + \frac{D_3}{t^3}, \quad (6)$$

where with $D_0 = 2 + \frac{64L^2}{\mu^2}$, we define

$$D_1 = \frac{16\sigma^2(1 + \frac{E^2}{n})}{\mu^2},$$
$$D_2 = \frac{256EL^2\sigma^2 \sum_{\tau=1}^{E-1}\sum_{j=0}^{\tau-1} D_0^j}{\mu^4},$$
$$D_3 = \frac{1028EL^2\sigma^2 \sum_{\tau=1}^{E-1}\sum_{j=0}^{\tau-1} D_0^j}{\mu^4}.$$

We note that relaxing to $E > 1$ does not fundamentally change the convergence behavior of CLG-SGD – the convergence behavior of $\mathcal{O}(1/(nt))$ is maintained. In addition, as stated at the beginning of this section, we can extend Theorem 2 for $K > 1$ and obtain the SGD noise term $\left(K^2 + E^2/n\right)$, which generalizes the $(1 + 1/n)$ noise term in C_0 of Theorem 1. In fact, Theorem 2 can recover Theorem 1 if we keep the same stepsize configuration and set $E = 1$.

B. Non-convex Loss Functions

Non-convex loss functions are often used in training deep neural networks. We now analyze the convergence behavior of CLG-SGD with non-convex loss functions, i.e., we remove Assumption 2 from the analysis. We note that for non-convex loss functions, it is well-known that SGD may converge to a local minimum or saddle point, and it is a common practice to evaluate the expected gradient norms as an indicator of convergence. In particular, an algorithm achieves an ϵ-suboptimal solution if

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \epsilon, \quad (7)$$

which guarantees the convergence to a stationary point [11].

Theorem 3 *Suppose Assumptions 1 and 3 hold. When the stepsize is set as $\eta_t = \gamma_t = \frac{1}{L\sqrt{T}}$, the convergence of CLG-SGD with $E = 1$ and non-convex loss functions satisfies*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{2L(f(x_0) - f^*)}{\sqrt{T}} + \frac{\sigma^2}{\sqrt{T}}\left(1 + \frac{1}{n}\right).$$

The asymptotic convergence can be directly obtained from Theorem 3 as $\mathcal{O}(1/\sqrt{nT})$., which matches the known result for local SGD without server training [14]. However, we note that a comparison to local SGD under the same configuration is much more involved than the case of strongly convex loss function. Fundamentally, this is due to the objective function in Eqn. (7), which does not guarantee convergence to the same (sub)optimal model. A detailed examination has revealed that it is possible to prove CLG-SGD outperforms local SGD in the early phases when both start at the same initial model. However, they may then converge to different stationary points as the learning process advances, making it impossible to analytically comparing their convergence behavior. We will examine their performances via experiments in Section V.

Theorem 4 reaffirms that the same $\mathcal{O}(1/\sqrt{nT})$ convergence rate under non-convex loss function can be maintained for $E > 1$.

Theorem 4 *Suppose Assumptions 1 and 3 hold. When the stepsize is set as $\eta_t = \gamma_t = \frac{1}{LE\sqrt{T}}$ and when*

$$T \geq \frac{4}{(\sqrt{n^2 + 4} - n)^2},$$

the convergence of CLG-SGD with non-convex loss functions satisfies

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x_t)\|^2 \leq \frac{2L(f(x_0) - f^*)}{\sqrt{T}} + \frac{\sigma^2(E-1)(2E-1)}{6nE^2T} + \frac{\sigma^2}{\sqrt{T}}\left(\frac{1}{n} + \frac{1}{E^2}\right).$$

We note that Theorem 4 recovers Theorem 3 if we set $E = 1$. However, the additional requirement $T \geq 4/(\sqrt{n^2 + 4} - n)^2$ has to be enforced, while Theorem 3 does not have any constraint on T, suggesting that it is a tighter bound for $E = 1$.

V. Experimental Results

A. Setup

Objectives. We have two goals in the experiments. First, we would like to corroborate the theoretical analysis in Section IV for CLG-SGD and compare with local SGD. To do this, we construct a data center setting in the experiment, where all parties can access the same dataset, and focus on the CIFAR-10 image classification task. In this setting, at each round, all the clients and the server sample a mini-batch of data samples from the dataset independently and

uniformly at random. We vary the number of clients as $n \in \{2, 4, 8, 16\}$, local SGD iterations as $E = \{4, 8\}$, and global SGD iterations as $K = \{4, 8\}$.

Second, we would like to evaluate the performance of CLG-SGD in applications that are beyond data center. For this, we choose to implement the cross-device federated learning setting in [6] with MNIST and CIFAR-10. The total dataset $\mathcal{D}$ for each task is partitioned (IID or non-IID) and stored in different parties (clients and server) before model training. We note, however, that we always assume the server data shares the same distribution as the total dataset, i.e., $\mathcal{D}_s$ is always sampled IID from the total dataset with the true distribution ν. This is necessary because otherwise the bias in the server dataset would always derail the model convergence, no matter how well local SGD performs at the clients. We represent the size of server dataset as GD, which is defined as the ratio of the actual server dataset size over the local dataset size at each client. We assume that each client has the same size of local dataset, and focus on varying GD at the server to evaluate the impact of large or small server datasets on CLG-SGD. We also vary the SGD iterations K on the server.

Baselines. We compare CLG-SGD against the following baseline methods: (1) **Server only**: server trains the ML model by only using server training data. Note that this case has no communication, and the plotted convergence against communication rounds should be interpreted as the corresponding SGD steps (i.e., the model is evaluated every E iterations). (2) **Local SGD** [5]: this is also known as FEDAVG in FL [6]. (3) **Local SGD+**: an enhanced version of local SGD/FedAvg that treats the server as an additional client. In other words, we add one more "client" that also participates local SGD/FedAvg. Note that the aggregation weight becomes unequal when the datasets are imbalanced, as discussed in [6].

B. Results for SGD Theory

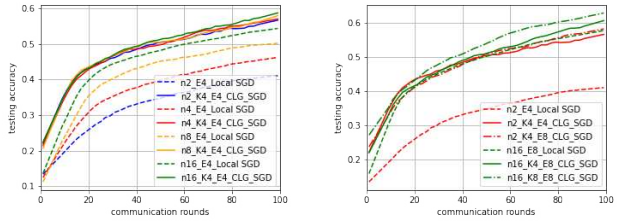

Fig. 1. CLG-SGD vs local SGD.

The experiment setting for Fig. 1 largely matches the requirement of Theorems. 3 and 4, which have established the same $\mathcal{O}(1/\sqrt{nT})$ convergence rate as local SGD but did not prove the performance advantage of CLG-SGD. We now see from the experiment results in Fig. 1(a) that CLG-SGD substantially improves the convergence over local SGD, not only in terms of the final test accuracy but also the convergence rate. The former also corroborates our conjecture that since the convergence for non-convex loss functions does not necessarily converge to the optimal model, different methods may converge to different stationary points. We also see that as n increases, the gain of CLG-SGD over local SGD diminishes, which is as expected since larger n boosts the advantage of parallelism over the cascaded server SGD.

We further vary K with n and evaluate the impact on the performance of CLG-SGD. Fig. 1(b) shows that for small number of clients ($n = 2$), the gain of CLG-SGD over local SGD is substantial, but further increasing the server SGD iterations K only brings marginal benefit. This, however, is not true for larger $n = 16$, where increasing K leads to more notable gain.

C. Results for Federated Learning

We now evaluate CLG-SGD in the federated learning framework. Local SGD, CLG-SGD and server-only are compared for various tasks. One difference to the previous experiment is that we now allow clients and server to only access the data samples that are locally stored (hence satisfying the requirement of FL). We also normalize E and K to represent the epochs of model training on the corresponding dataset, which is proportional to the SGD steps when we fix the mini-batch size.

MNIST. When the local datasets are IID, Fig. 2 shows that even when the server has 20% of total training samples (which is a large amount for the MNIST dataset), training at the server alone does not have the same accuracy (on the validation set) as local SGD and CLG-SGD. On the other hand, we see clearly that CLG-SGD with a modest choice of $K = 3$ (purple line) achieves the best performance that has more than $3\times$ convergence improvement over the standard local SGD/FedAvg, while increasing the value to $K = 5$ (brown line) is more beneficial at the beginning of training but plateaus towards the end. Local SGD+ also has improved performance over local SGD, which is not surprising since it has one more "super client" that has significant amount of data. Nevertheless, this gain is not as good as CLG-SGD.

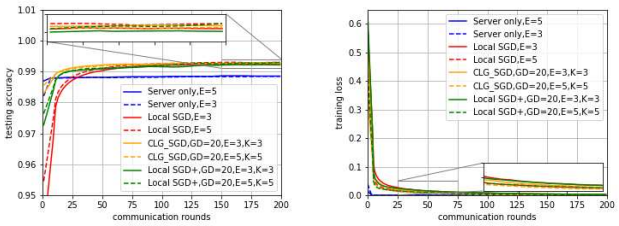

Fig. 2. IID MNIST. Left: test accuracy; Right: training loss.

If the dataset is non-IID at clients (but server still has an IID dataset), CLG-SGD has very noticeable advantage over other methods as shown in Fig. 3. If the clients only contain non-IID data samples, the performance of local SGD drops drastically from the IID case. Local SGD+, with the addition

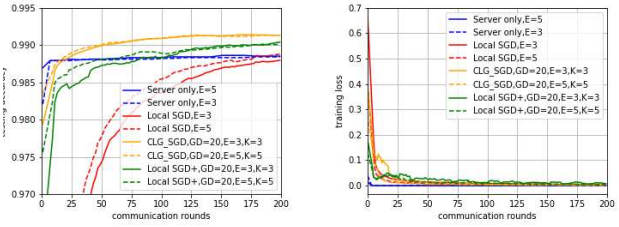

Fig. 3. Non-IID MNIST. Left: test accuracy; Right: training loss.

of a "super client", can recover some of the losses due to non-IID data partition. However, even with a limited utilization of server training (e.g., $K = 3$), CLG-SGD already outperforms all of the local SGD-based algorithms. A larger utilization of server training would further improve the accuracy and move it closer to the IID performance.

CIFAR-10. This is a much harder task than MNIST. When the server has a large portion of the total dataset, Fig. 4 shows that the server-only baseline is not as bad as in the MNIST task. Nevertheless, CLG-SGD still has the best overall performance for the case of IID local dataset partitioning. However, for the case of non-IID, the large dataset at the server leads to dominating performance in the early stages of model training, which diminishes the potential gain of local SGD and CLG-SGD – we still see from Fig. 5 that CLG-SGD with $GE = 5$ has the best overall performance, but the gain is marginal.

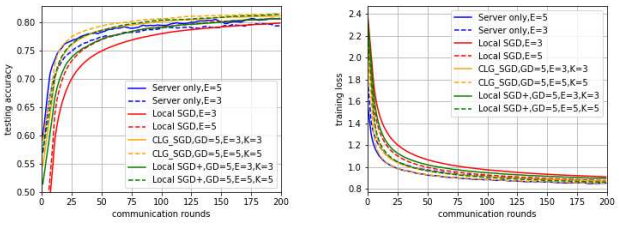

Fig. 4. IID CIFAR-10. Left: test accuracy; Right: training loss.

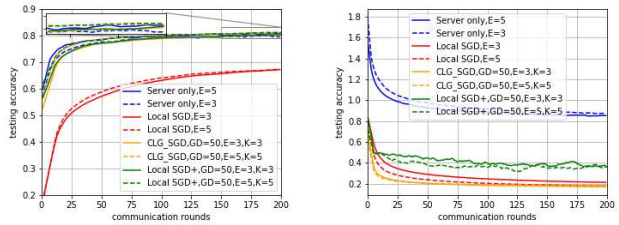

Fig. 5. Non-IID CIFAR-10. Left: test accuracy; Right: training loss.

VI. Conclusions

Based on a simple observation that the parameter server typically has, large or small, some amount of data that can be used to help federated learning, we have analyzed the convergence performance of a cascading local and global SGD design that naturally combines the benefits of both centralized and distributed SGD. This cascading structure complicated the analysis of CLG-SGD, and we have proved the convergence behavior under both strongly convex and non-convex loss functions. The advantage of CLD-SGD over vanilla local SGD was established when the stepsizes were chosen properly. Extensive experiments using standard MNIST and CIFAR-10 datasets were carried out, which not only corroborated the theoretical analysis but also empirically demonstrated the performance advantage of CLG-SGD over various baselines.

References

[1] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[2] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–33, 2020.

[3] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. a. Ranzato, A. Senior, P. Tucker, K. Yang, Q. Le, and A. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012.

[4] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *USENIX OSDI*, October 2014, pp. 583–598.

[5] S. U. Stich, "Local SGD converges fast and communicates little," in *Proc. ICLR*, 2018.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th AISTATS*, Apr. 2017, pp. 1273–1282.

[7] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Advances in Neural Information Processing Systems*, 2018, pp. 2525–2536.

[8] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization," *arXiv:1905.03817*, 2019.

[9] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," in *SysML Conference*, 2019.

[10] D. A. E. Acar, Y. Zhao, R. Matas, M. Mattina, P. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization," in *International Conference on Learning Representations*, 2021.

[11] J. Wang and G. Joshi, "Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms," in *ICML Workshop on Coding Theory for Machine Learning*, 2019.

[12] 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*, 2020.

[13] Z. Li and P. Richtárik, "A unified analysis of stochastic gradient methods for nonconvex federated optimization," *arXiv:2006.07013*, 2020.

[14] 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 *Proc. AAAI*, vol. 33, 2019, pp. 5693–5700.