A Convergence Theory for Federated Average: Beyond Smoothness

Xiaoxiao Li  
Department of ECE  
UBC  
Vancouver, Canada  
xiaoxiao.li@ece.ubc.ca

Zhao Song  
Adobe Research  
San Jose, USA  
zsong@adobe.com

Runzhou Tao  
Department of CS  
Columbia University  
New York, USA  
runzhou.tao@columbia.edu

Guangyi Zhang  
Department of ECE  
McGill University  
Montreal, Canada  
guangyi.zhang@mail.mcgill.ca

Abstract—Federated learning enables a large amount of edge computing devices to learn a model without data sharing jointly. As a leading algorithm in this setting, Federated Average (FedAvg), which runs Stochastic Gradient Descent (SGD) in parallel on local devices and averages the sequences only once in a while, have been widely used due to their simplicity and low communication cost. However, despite recent research efforts, it lacks theoretical analysis under assumptions beyond smoothness. In this paper, we analyze the convergence of FedAvg. Different from the existing work, we relax the assumption of strong smoothness. More specifically, we assume the semi-smoothness and semi-Lipschitz properties for the loss function, which have an additional first-order term in assumption definitions. In addition, we also assume bounded gradient assumption in the convergence analysis scheme. As a solution, this paper provides a theoretical convergence study on Federated Learning.

Index Terms—Federated learning, semi-smoothness, no-critical-point, semi-Lipschitz.

I. INTRODUCTION

With the growing of computational power on edge devices, such as mobile phones, wearable devices, smart watches, self-driving cars, and so on, developing distributed optimization methods to address the needs of those applications is increasingly demanded. There are three core challenges existing in the distributed computing applications, including expensive communication, privacy concerns, and heterogeneity. To tackle the above-mentioned challenges, federated learning (FL) has emerged as an important paradigm in today’s machine learning for distributed learning that enables different clients (also known as nodes) to collaboratively learn a model while keeping their private data. To train an FL algorithm in a distributed manner, the clients must transmit their training parameters to a central server. Typically, the central server has the same model architecture as the local clients. Similar to centralized parallel optimization, FL lets the clients do most of the computation while the central server updates the model parameters using the descending directions returned by the local clients.

However, learning with FL significantly differs from the traditional parallel optimization in distributed learning in the various needs, including piracy requirements, large-scale machine learning and efficiency. To meet these unique requirements, the most popular existing and easiest to implement FL strategy is Federated Average (FedAvg) [2], where clients collaboratively send updates of locally trained models to a global server. Each client runs a local copy of the global model on its local data. The global model’s weights are then updated with an average of local clients’ updates and deployed back to the clients. This strategy builds upon previous distributed learning work by supplying local models and performing training locally on each device. Hence FedAvg potentially empowers clients (especially clients with small datasets) to collaboratively learn a shared prediction model while keeping all training data locally.

Although FedAvg has shown successes in classical Federated Learning tasks, it suffers from slow convergence and low accuracy in most non-iid contents [3], [4]. There have been many efforts developing convergence guarantees for FL algorithms, i.e., how the convergence rate is affected by the client local update epochs, how many communication rounds are required to achieve a targeted model performance. There have been many efforts developing convergence guarantees for FL algorithms [5], [6], [7], [8] on FedAvg.

Yet, the optimization and convergence analysis in FL is quite non-trivial. On the one hand, the optimized objective are usually not only non-convex but even non-smooth. For example, mapping functions may have non-linear operations, such as ReLU activation and maxing out the label for objective functions. On the other hand, different from centralized training, each client’s local update steps and the difference between local and global model (aka local drift) need to be considered.

In this paper, we study the following fundamental question:

Can we show that FedAvg converges under mild assumptions?

To answer the above question, the following two challenges need to be addressed: 1) How to find suitable assumptions and 2) How to design a framework to handle local updates and local drift. Inspired by overparameterized optimization theory, we introduce the semi-smooth and no-critical-point properties. We develop a new framework to reason about the convergence of federated learning. Our proof solves the key task of bounding the local drift under the semi-smoothness assumption.
We provide a conclusion for the paper and discuss possible data distribution.

Clients in each round of communication, and heterogeneous efficiency, understanding the effect of sampling a subset of strategies [22], [23], [24], [25], [26]. Existing studies on federated learning involve learning a machine learning model to be trained on local clients in a distributed fashion. An essential bottleneck in such a distributed training algorithms. Under our valid assumptions, the local drift in FedAvg is appropriately bounded when the local updating learning rate is controlled by the parameters associated with the assumptions.

Our theoretical results indicate how local updates steps and learning rate affect the number of communication rounds in FedAvg.

b) Organization: The paper organization is as follows. In Section II, we discuss related work about FL algorithms, especially variants of FedAvg, and its convergence analysis under different settings. In Section III, we describe our problem setting that we consider FedAvg algorithm and state the semi-smoothness, no-critical-point and semi-Lipschitz assumptions to be used in our proof. In Section IV, we elaborate on the rationality of making the three assumptions and state our results on convergence of FedAvg under these assumptions. In Section V, we give a proof sketch of our results first on a simplified non-FL case, and then on the FL case. In Section VI, we provide a conclusion for the paper and discuss possible future work and the social impact of this work.

II. RELATED WORK

a) Federated learning: With the growth of computational power, data are massively distributed over an incredibly large number of devices. Federated learning is proposed to allow machine learning models to be trained on local clients in a distributed fashion. An essential bottleneck in such a distributed training on the cloud is the communication cost. Federated average (FedAvg) [2] firstly addressed the communication efficiency problem. FedAvg algorithm allows devices to perform local training of multiple epochs to reduces the number of communication rounds, then average model parameters from the client devices. Later, a myriad of variations and adaptations have arisen [9], [10], [11], [12], [13], [14], [15].

As stated earlier, federated learning involves learning a centralized model from distributed client data. This centralized model benefits from all client data and can often result in a beneficial performance e.g. in including next word prediction [16], [17], emoji prediction [18], vocabulary estimation [19], and predictive models in health [20], [21]. Multiple research efforts studying the issues on more efficient communication strategies [22], [23], [24], [25], [26]. Existing studies on federated learning have mostly focused on improving communication efficiency, understanding the effect of sampling a subset of clients in each round of communication, and heterogeneous data distribution.

b) Convergence of FedAvg: For identical clients, FedAvg coincides with parallel SGD analyzed by [27] who proved asymptotic convergence. [28] and, more recently [29], [30], [31], gave a sharper analysis of the same method, under the name of local SGD, also for identical functions. The analysis of FedAvg is more sophisticated than parallel SGD due to local drift, which represents the difference between local and global models. The divergence is empirically observed in [10] on non-iid data. Some analyses constrain this drift by assuming a bounded gradient [7], [6]. Although a few recent studies [32], [33] do not require bounded gradient, their theories are limited to L-smoothness assumption. Specifically, [32] only discusses the convergence on convex cases and [33] relax the bound to grow with gradient norm for non-convex cases. Alternatively [31] treat the drift as additional noise. In recent work, [8] proposes to reduce the gradient diversity, where authors suggest augmenting the local gradients with a controlled variance.

III. PRELIMINARY

In this section, we introduce the setting of our problem. We first introduce the notations to be used throughout the paper. Then, we formulate our learning model by defining the local and total loss function and explaining our FedAvg algorithm. Finally we state the three non-smooth assumptions, namely semi-smoothness, no-critical-point and semi-Lipschitz, based on which we prove FedAvg’s convergence.

a) Notations: For any positive integer n, we use [n] to denote set \{1, 2, · · · , n\}. For a vector x, we use \|x\|_2 to denote its l_2 norm. For a matrix W, we use \|W\|_F to denote the spectral norm of W. We use \mathbb{E}[\cdot] to denote the expectation of a random variable if its expectation is existing. We use \Pr[\cdot] to denote the probability.

Let n denote the number of input data points. Let N denote the number of clients. We can think of each client will have n/N data points. Let S_1 ∪ S_2 ∪ · · · ∪ S_N = [n] and S_i ∩ S_j = ∅. Given n input data points and labels \{(x_1,y_1), (x_2,y_2), · · · , (x_n,y_n)\} ∈ \mathbb{R}^d × \mathbb{R}.

b) Problem formulation: In this work, we consider the following federated learning model using FedAvg algorithm. Suppose N clients are in the federated learning system. We define the local loss function L_c of c-th client for c ∈ [N], L_c(W, x) = \frac{1}{2} \sum_{i \in S_c} \text{loss}(x_i,y_i)^2, where c-th client holds training data \{(x_i,y_i) | i \in S_c\}, loss can be l_2 loss, cross entropy loss, and others in practice. When all clients are activate, the total loss function is defined as L(W, x) = \frac{1}{N} \sum_{c=1}^N L_c(W, x). We formalize the problem as minimizing the sum of loss functions over all clients: \min_{W \in \mathbb{R}^{d \times M}} L(W).

We define \gamma_c(W) := \nabla L_c(W; \zeta_c) be an unbiased stochastic gradient of L_c with variance bounded by \sigma^2.

c) Algorithm: A typical implementation of FedAvg contains an additional global model (with the same architecture as a local model) and performs in the following way. First, the local client (say the c-th) trains the local neural network and updates local model weight W_c. Then, the local model weight W_c is sent to the global model. Later, the global model average
We state the assumptions

Assumption III.1. We state the assumptions

- Semi-smoothness, Section IV-A

\[ L(W) \leq L(U) + \langle \nabla L(U), W - U \rangle + b\|W - U\|^2 + a\|W - U\| \cdot L(U)^{1/2}. \]

- No critical point, Section IV-B

\[ \tau_1^2 L(U) \leq \|\nabla L(U)\|^2 \leq \tau_2^2 L(U). \]

- Semi-Lipschitz, Section IV-C

\[ \|\nabla L(W) - \nabla L(U)\|^2 \leq \beta^2 \|W - U\|^2 + \alpha^2 \|W - U\| L(U)^{1/2}. \]

IV. Framework Going Beyond Smoothness

In this section, we give formal definitions of the three conditions we use to prove the convergence of \texttt{FedAvg}. We relax the smoothness condition in the classical analysis to semi-smoothness which is shown to be held by neural networks. No critical point condition weakens the bounded gradient property that are widely used in previous convergence analysis on federated learning algorithms. And the semi-Lipschitz condition weakens the classical Lipschitz property.

A. Smoothness property

To ensure the objective function decreases over training time, one relies on the smoothness property in classical optimization theory. We first start with describing the definition of \( \beta \)-smoothness in FL, which is extended from the classical analysis.

Definition IV.1 (\( \beta \)-smoothness). For any function \( L \), we say it is \( \beta \)-smooth if for any \( W, U \)

\[ L(U) \leq L(W) + \langle \nabla L(W), U - W \rangle + \frac{\beta}{2}\|W - U\|^2. \]

However, the neural networks, the widely used model in FL, may not meet the twice differentiability requirement of \( \beta \)-smoothness (i.e., the ReLU activation). Thus, a milder assumption of smoothness is often required. To deal with the issue, in [34], semi-smoothness is proposed and shown to be held by neural networks. To extend the semi-smoothness definition in FL, we have:

Definition IV.2 ((\( \alpha, \beta \))-semi-smoothness). For any function \( L \), we say it is \((\alpha, \beta)\)-semi-smooth if for any \( W, U \)

\[ L(U) \leq L(W) + \langle \nabla L(W), U - W \rangle + b\|W - U\|^2 + a\|W - U\| \cdot L(W)^{1/2}. \]

It is worth noting that, different from the smoothness definition, we have an additional first order term \( \|W - U\| \) on the right hand side.

B. No critical point

Finding approximate critical points of a non-smooth and non-convex function was challenging[35], until [34] proof that the gradient bounds for points that are sufficiently close to the random initialization. It is proved in [34] that there is no critical point for square loss function for neural networks. Thus the no critical point (Theorem 3 in [34]) is a property in the nature of neural networks in the classical training regime. Therefore, we extend the definition of the no critical point property to FL as:

Definition IV.3 (No critical point). Let \( \mathcal{U} \) be a neighbor set of \( U^\ast \) (a minimum of \( L \)). We say there is no critical point for a function \( L \), there exist constants \( 0 < \tau_1 < 1 \) and \( \tau_2 > 0 \), if for any \( U \in \mathcal{U} \), \( \tau_1^2 L(U) \leq \|\nabla L(U)\|^2 \leq \tau_2^2 L(U) \).

Definition IV.3 shows that the gradient norm is large when the objective function is large. This means that there are no saddle points or critical points when we are sufficiently close to the random initialization. Thus, we hold a good brief of finding global minima of the objective function.

Bounded gradient property is a popular scheme for convergence analysis used in some of the previous work [36], [6], which is defined as:

Definition IV.4 (Bounded gradient [4]). We say a function has bounded gradient, if there exists \( G \geq 0 \) such that for any \( U \), \( \|\nabla L(U)\|^2 \leq G^2 \) holds.

The no critical point assumption is a weaker assumption for both strong convexity and bounded gradient.

C. Lipschitz property

In the classical analysis, the \( \beta \)-Lipschitz defined in Definition IV.5 is assumed. We also have the \( \beta \)-smoothness defined in Definition IV.1 is implied by the \( \beta \)-Lipschitz.

Definition IV.5 (\( \beta \)-Lipschitz). For a function \( L \), we say it is \( \beta \)-smooth if for any \( W, U \)

\[ \|\nabla L(W) - \nabla L(U)\| \leq \beta \cdot \|W - U\|. \]

[34] shows that for the overparameterized neural networks, the first-order term is much smaller than the second-order term during neural networks evolution. In this case, Definition IV.2 ((\( \alpha, \beta \))-semi-smoothness) is close to, but still not interchangeable with the classical Lipschitz smoothness. We propose to consider a weaker definition of Lipschitz in FL, due to the multiple local client update steps. The milder assumption is defined as \((\alpha, \beta)\)-semi-Lipschitz.

Definition IV.6 ((\( \alpha, \beta \))-semi-Lipschitz). For a function \( L \), we say it is \((\alpha, \beta)\)-semi-smooth if for any \( W, U \)

\[ \|\nabla L(W) - \nabla L(U)\|^2 \leq \beta^2 \cdot \|W - U\|^2 + \alpha^2 \cdot \|W - U\| \cdot \max\{L(W)^{1/2}, L(U)^{1/2}\}. \]

Since the semi-Lipschitz definition should be symmetric in terms of \( U \) and \( W \), we use \( \max\{L(W)^{1/2}, L(U)^{1/2}\} \) instead of \( L(W)^{1/2} \) as in Definition IV.2.
**D. Our results**

Standard global convergence analysis for gradient descent uses smoothness and strong convexity. We show the relaxed assumptions, \((a,b)\)-semi-smoothness and \((\tau_1, \tau_2)\)-no critical point, are sufficient to analyze the convergence for gradient descent. To derive the convergence for FedAvg, we also need to bound local gradient updates, which requires the semi-Lipschitz property.

Let \(U^\ast\) denote a minimizer of \(L\). Under Assumption IV.2, IV.3 and IV.6, we state our result and provide a proof sketch (see Section V-B).

**Theorem IV.7** (Main result). *If the loss functions \(L_c (c \in [N])\) for each client \(c\) is: \((a,b)\)-semi-smooth, \((\alpha, \beta)\)-semi-Lipschitz, \((\tau_1, \tau_2)\)-non-critical point, for any parameters \(\eta\) less than \[
\min\{1/(\alpha^2 K), 1/(10^2 K \tau_2 (\beta + \alpha)), 1/(10^2 \sqrt{K} (\beta + \alpha))\},
\]

and \(\eta < \tau_1^2/(20K\eta_l)\),

for FedAvg, we have \[
\mathbb{E}[L(U^r) - L(U^\ast)] \leq (1 - \lambda_1)^r \cdot (L(U^0) - L(U^\ast)) + 2\lambda_2,
\]

where
\[
\lambda_1 = \frac{K\eta \eta_l}{4} (1 - 4bK\eta_l - 2a)\tau_1^2,
\]
\[
\lambda_2 = (1 + a + bK\eta_l)\frac{K\eta \eta_l}{10} a^2.
\]

**Corollary IV.8.** *For any desired \(\epsilon\), using a step-size of \(\eta \leq \min\{\tau_1^2/(20K\eta_l), 2\epsilon/(\alpha^2(1 + a + \tau_1^2/20K))\},

after rounds \(R = \log\left(\frac{2(L(U^0) - L(U^\ast))}{\epsilon}\right)/\lambda_1\), we have \(\mathbb{E}[L(U^r) - L(U^\ast)] \leq \epsilon\).

Given the definition of \(\lambda_1\), we notice the trade-off that either a small or a large \(K\) will result a large \(R\) under the assumptions.

**V. PROOF SKETCH**

In this section, we show a proof sketch of our main result Theorem IV.7. To better illustrate the proof structure, we first prove the convergence of the non-federated learning gradient descent algorithm under the semi-smoothness and non-critical-point conditions. The simplified proof shares the same structure as the final proof. Then, we consider the federated learning case with four steps, including the key step of bounding local drift, and discuss the difference between FL and non-FL cases.

**A. Non-federated learning case (simplified)**

**Proposition V.1.** *Let \(x^\ast\) denotes a minimum of \(L\). Suppose we run gradient descent algorithm to update \(x_{t+1}\) in each iteration as follows: \(x_{t+1} = x_t - \eta \cdot \nabla L(x)|_{x=x_t}\). If the loss function \(L\) is \((a,b)\)-semi-smooth, \((\tau_1, \tau_2)\)-non-critical point, \(0.5\tau_1^2 \geq a\tau_2\), using \(\eta \leq \tau_1^2/(10b\tau_2^2)\) then we have \[
L(x_{t+1}) - L(x^\ast) \leq (1 - \lambda)(L(x_t) - L(x^\ast))
\]

where \(\lambda = 0.1\eta\tau_1^2\).

See the proof in [1].

**B. Federated learning case, Theorem IV.7**

In this section, we show a proof sketch for Theorem IV.7. We show the simplified proof with gradient descent. The complete proof with stochastic gradient descent is presented in the full version [1].

The first step in our proof is using the semi-smoothness of \(L\) to compute \[
L(U^{r+1}) - L(U^\ast) \leq L(U^r) - L(U^\ast) + \langle \nabla L(U^r), \Delta U^r \rangle + b\|\Delta U^r\|^2 + a\|\Delta U^r\| \cdot L(U^\ast)^{1/2}
\]

The gradient update \(\Delta U^r\) is an average of local gradients. Unlike the centralized case, the terms \(\langle \nabla L(U^r), \Delta U^r \rangle\) and \(\|\Delta U^r\|\) cannot be simply bounded by \((\tau_1, \tau_2)\)-non-critical point assumption.

Next, we are going to show the bounds for the RHS. The remained proof is organized as follows:

**Bounding \(\langle \nabla L(U^r), \Delta U^r \rangle\).** In the non-federated case, we have \(\langle \nabla L(U^r), \Delta U^r \rangle = -\eta\|\nabla L(U^r)\|^2\). In the federated case, \(\Delta U^r\) is an average of local gradients, but we need an upper bound for \(\langle \nabla L(U^r), \Delta U^r \rangle\) in terms of global parameter \(U^r\) rather than local parameters \(W^r,c\). Thus, we upper bound \(\langle \nabla L(U^r), \Delta U^r \rangle\) by \(-0.5\eta\|\nabla L(U^r)\|^2\) plus differences of gradients of loss at local and global parameters, i.e. \(\|\nabla L_c(W^r,c,k) - \nabla L_c(U^r)\|^2\).

To bound \(\|\nabla L_c(W^r,c,k) - \nabla L_c(U^r)\|^2\), we need the Lipschitz property, which is not used in the non-federated case. We define \(\xi = 1/KN \sum_{k=1}^K \sum_{c=1}^N \|W^r,c,k - U\|^2\).

Then, \(\langle \nabla L(U^r), \Delta U^r \rangle\) is upper bound by a sum of \(-0.5\eta\|\nabla L(U^r)\|^2, \xi\) and \(L(U^r)\).

**Bounding \(\|\Delta U^r\|^2\).** The same as the non-federated case, we use non-critical point property to bound the norm of gradients, i.e. for all \(k \in [K]\), \(\|W^r,c,k\|^2 \leq \tau_2^2 L(W^r,c,k)\).

Then we use the result of the non-federated case, for all \(k \in [K]\), \(L(W^r,c,k) \leq L(U^r)\).

**Bounding \(\xi\).** The term \(\xi\) is an average of local drifts caused by the local updates. It is expected that this drift is small for overparameterized neural network. We upper bound \(\xi\) by induction.

**Choosing parameters.** Finally, we put all the bounds together and explain how to choose parameters to assure decrease in loss for each global training round.

We consider gradient descent in this proof sketch for the ease of presentation, the proof for stochastic gradient descent is shown in the full version [1]. The gradient update for FedAvg is \[
U^{r+1} = U^r + \Delta U^r
\]

\[
= U^r - \frac{\eta}{KN} \sum_{k \in [K], c \in [N]} \nabla L_c(W^r,c,k-1),
\]
where $W_{c,k}$ is defined by

$$W_{c,0} = U_r, \quad W_{c,k} = W_{c,k-1} - \eta_k \nabla L_c(W_{c,k-1}),$$

where $\bar{\eta} = K \eta_n$. The effective step.

**a) Bounding $\langle \nabla L(U^r), \Delta U^r \rangle$:** To simplify the notation, we ignore the superscript $r$. First, we compute the term $\langle \nabla L(U), \Delta U \rangle$ in Eq. (1).

$$\langle \nabla L(U), \Delta U \rangle = -\bar{\eta} \langle \nabla L(U), 1 \rangle - \frac{1}{K} \sum_{k e [K]} \nabla L_c(W_{c,k-1})$$

$$\leq -\frac{\bar{\eta}}{2} \langle \nabla L(U) \rangle + \frac{1}{2K} \sum_{k e [K]} \nabla L_c(W_{c,k-1}) - \nabla L(U) \rangle^2$$

$$= -\frac{\bar{\eta}}{2} \langle \nabla L(U) \rangle + \frac{1}{2K} \sum_{k e [K]} \nabla L_c(W_{c,k-1}) - \nabla L(U) \rangle^2$$

$$\leq -\bar{\eta} \langle \nabla L(U) \rangle + \frac{1}{2K} \sum_{k e [K]} \nabla L_c(W_{c,k-1}) - \nabla L(U) \rangle^2,$$

where second step follows from $-ab = \frac{1}{2}((b-a)^2 - a^2) \leq \frac{1}{2}(b^2 - a^2)$, the third step follows from

$$\nabla L(U) = \frac{1}{K} \sum_{k e [K]} L_c(U),$$

and the last step follows from $n \sum_{i=1}^n a_i^2 \leq n \sum_{i=1}^n a_i^2$.

To bound the difference of gradients of loss, one have to use the Lipschitz property as discussed in section IV.C. By $(\alpha, \beta$)-semi-Lipschitz and $2ab \leq a^2 + b^2$, we have

$$\langle \nabla L_c(W_{c,k-1}) - \nabla L_c(U) \rangle^2$$

$$\leq \beta^2 \langle W_{c,k-1} - U \rangle^2 + \alpha^2 \langle W_{c,k-1} - U \rangle \langle L_c(U) \rangle$$

$$\leq (\beta^2 + \alpha^2) \langle W_{c,k-1} - U \rangle^2 + \alpha^2 \langle L_c(U) \rangle$$

Combining the definition of $\xi$ in (2) and Eq. (3), we have

$$\langle \nabla L(U), \Delta U \rangle$$

$$\leq -\frac{\bar{\eta}}{2} \langle \nabla L(U) \rangle + \frac{2\beta^2 + \alpha^2}{4} \eta \cdot \xi + \frac{\alpha^2}{4} \eta \cdot L(U)$$

$$\leq -\frac{2\tau_1^2 - \alpha^2}{4} \eta \cdot L(U) + \frac{2\beta^2 + \alpha^2}{4} \eta \cdot L(U),$$

where the second step follows from the non-critical point property.

**b) Bounding $||\Delta U||^2$:** Next, we consider the term $||\Delta U||^2$ in Eq. (1).

$$||\Delta U||^2 \leq \frac{\bar{\eta}^2}{2K} \sum_{k e [K]} \langle \nabla L_c(W_{c,k-1}) \rangle^2$$

$$\leq \frac{\tau_1^2}{2} \sum_{k e [K]} L_c(W_{c,k-1}),$$

where the first step follows from $\sum_{i=1}^n a_i^2 \leq n \sum_{i=1}^n a_i^2$, and the second step follows from the non-critical point property.

By applying Proposition V.1 to $L_c(W_{c,k-1})$, for any positive integer $k \geq 2$, we have

$$L_c(W_{c,k}) \leq L_c(W_{c,k-1})$$

$$\leq \cdots \leq L_c(U)$$

Combining Eq. (6) and (7), $||\Delta U||^2$ is upper bounded as

$$||\Delta U||^2 \leq \frac{\tau_1^2}{2} \eta \cdot L(U).$$

**c) Bounding $\xi$:** Then, we upper bound $\xi$ by induction. For $k = 1$, $\xi = 0$, and for $k \geq 1$, we compute one step local update.

$$||W_{c,k+1} - U||^2$$

$$= ||W_{c,k} - U - \eta \cdot \nabla L_c(W_{c,k})||^2$$

$$\leq (1 + \frac{1}{K}) \cdot ||W_{c,k} - U||^2 + \frac{\tau_1^2}{2} \eta \cdot \nabla L_c(W_{c,k})$$

$$\leq (1 + \frac{2}{K}) \cdot ||W_{c,k} - U||^2 + \frac{\tau_1^2}{2} \eta \cdot \nabla L_c(W_{c,k})$$

$$\leq (1 + \frac{2}{K}) \cdot ||W_{c,k} - U||^2 + \frac{\tau_1^2}{2} \eta \cdot L_c(U),$$

where the second step follows from $2ab \leq a^2 + b^2$, the third step follows from $K \geq 2$, and the last step follows from the non-critical point property, and Eq. (7).

Unrolling the recursion above,

$$E[||W_{c,k} - U||^2] \leq (K \tau_2 \eta^2 \cdot L_c(U)) \cdot \sum_{i=1}^K (1 + \frac{2}{K})^{i-1}$$

$$\leq 4K \tau_2 \eta^2 \cdot L_c(U),$$

where the geometric sequence $\sum_{i=1}^K (1 + \frac{2}{K})^{i-1} \leq 4K$. Averaging over $c$ and $k$, we upper bound

$$\xi \leq 4K^2 \tau_2 \eta^2 \cdot \frac{1}{N} \sum_{c=1}^N L_c(U) = 4K^2 \tau_2 \eta^2 \cdot L(U).$$

**d) Putting together:** By combining Eq. (1), (5), (8), and (9), we have

$$L(U^{r+1}) - L(U^*)$$

$$\leq L(U^r) - L(U^*) - \frac{2\tau_1^2 - \alpha^2}{4} \eta \cdot L(U^r)$$

$$+ (2\beta^2 + \alpha^2) K^2 \tau_2 \eta^2 \cdot L(U^r)$$

$$+ b \tau_2 \eta^2 \cdot L(U^r) + a \tau_2 \eta \cdot L(U^r)$$

$$\leq (1 - A) L(U^r) - L(U^*),$$

where

$$A = \frac{2\tau_1^2 - \alpha^2}{4} \eta - a \tau_2 \eta - (2\beta^2 + \alpha^2) K^2 \tau_2 \eta^2 \eta - b \tau_2 \eta^2 .$$

**e) Choosing parameters:** We need to carefully tune the parameters to find a $\gamma > 0$, such that $A \geq \gamma$. By choosing

$$\alpha \leq 0.5 \tau_1, \quad a \tau_2 \leq 0.1 \tau^2,$$

$$\eta \leq \tau_1/(4KG(2\beta + \alpha)), \quad \eta \leq \tau_2^2/(16br^2),$$

we have

$$A \geq \frac{\tau_1^2}{8} \eta := \gamma.$$ (11)

By plugging (11) into Eq. (10), we get

$$cL(U^{r+1}) - L(U^*) \leq (1 - \gamma)cL(U^r) - L(U^*)$$

$$\leq (1 - \gamma)(L(U^r) - L(U^*)),$$

where the second step follows from $L(U^*) \geq 0$. 1296
In this paper, we analyze the convergence of FedAvg without the commonly-used smoothness assumption, and improve the theoretical convergence analysis of FedAvg under the non-convex and non-smooth settings. Under the non-smooth setting, it is challenging to make suitable assumptions. We introduce the semi-smoothness assumption and non-critical point assumption to tackle these problems. Besides, when considering FedAvg, local drift is usually difficult to analyze and bound. Under our new theoretical framework, local drift and the progress of gradient are bounded appropriately. By our milder assumptions, we can prove the convergence of FedAvg with the vanilla SGD update.

Our work sheds light on the theoretical understanding of FedAvg. In addition, in our paper, we show how to choose parameters for FedAvg, such as learning rate $\eta_1, \eta_2$, communication round $R$. We leave the detailed discussion about the effect of parameters in assumptions, like $\gamma_1, \gamma_2$, for future work. Although the results are promising, we should consider $\tau$ for

VI. DISCUSSION AND CONCLUSION

In this paper, we analyze the convergence of FedAvg with the vanilla SGD update.

REFERENCES

1. X. Li, Z. Song, R. Tao, and G. Zhang, “A convergence theory for federated average: Beyond smoothness,” arXiv preprint arXiv:2112.04977, 2019.

2. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Artificial Intelligence and Statistics. PMLR, 2017, pp. 1273–1282.

3. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” in Conference on Machine Learning and Systems, 2020a, 2020.

4. X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of fedavg on non-iid data,” arXiv preprint arXiv:1907.02189, 2019.

5. A. Khaled, K. Mishchenko, and P. Richtárik, “First analysis of local gd on heterogeneous data,” arXiv preprint arXiv:1909.04715, 2019.

6. 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, vol. 33, 2019, pp. 5693–5700.

7. S. Wang, T. Tsor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, “Adaptive federated learning in resource constrained edge computing systems,” IEEE Journal on Selected Areas in Communications, vol. 37, no. 6, pp. 1205–1221, 2019.

8. S. K. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, “Scaffold: Stochastic controlled averaging for federated learning,” in ICML. PMLR, 2020, pp. 5132–5143.

9. H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, “Federated learning with matched averaging,” arXiv preprint arXiv:2002.06440, 2020.

10. Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-iid data,” arXiv preprint arXiv:1806.00582, 2018.

11. 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.