FL-NTK: A Neural Tangent Kernel-based Framework for Federated Learning Analysis

Baihe Huang [1] **Xiaoxiao Li** [2] **Zhao Song** [3] **Xin Yang** [4]

Abstract

Federated Learning (FL) is an emerging learning scheme that allows different distributed clients to train deep neural networks together without data sharing. Neural networks have become popular due to their unprecedented success. To the best of our knowledge, the theoretical guarantees of FL concerning neural networks with explicit forms and multi-step updates are unexplored. Nevertheless, training analysis of neural networks in FL is non-trivial for two reasons: first, the objective loss function we are optimizing is non-smooth and non-convex, and second, we are even not updating in the gradient direction. Existing convergence results for gradient descent-based methods heavily rely on the fact that the gradient direction is used for updating. This paper presents a new class of convergence analysis for FL, Federated Learning Neural Tangent Kernel (FL-NTK), which corresponds to overparamterized ReLU neural networks trained by gradient descent in FL and is inspired by the analysis in Neural Tangent Kernel (NTK). Theoretically, FL-NTK converges to a global-optimal solution at a linear rate with properly tuned learning parameters. Furthermore, with proper distributional assumptions, FL-NTK can also achieve good generalization.

1. Introduction

In traditional centralized training, deep learning models learn from the data, and data are collected in a database on the centralized server. In many fields, such as healthcare and natural language processing, models are typically learned from personal data. These personal data are subject to regulations such as California Consumer Privacy Act (CCPA) (Legislature, 2018), Health Insurance Portability and Accountability Act (HIPAA) (Act, 1996), and General Data Protection Regulation (GDPR) of European Union. Due to the data regulations, standard centralized learning techniques are not appropriate, and users are much less likely to share data. Thus, the data are only available on the local data owners (i.e. edge devices). Federated learning (FL) is a new type of learning scheme that avoids centralizing data in model training. FL allows local data owners (also known as clients) to locally train the private model and then send the model weights or gradients to the central server. Then central server aggregates the shared model parameters to update new global model, and broadcasts the the parameters of global model to each local client.

Quite different from the centralized training, FL has the following unique properties. First, the training data are distributed on an astonishing number of devices, and the connection between the central server and the device is slow. Thus, the computational cost is a key factor in FL. In communication-efficient FL, local clients are required to update model parameters for a few steps locally then send their parameters to the server (McMahan et al., 2017). Second, due to the fact that the data are collected from different clients, the local data points can be sampled from different local distributions. When this happens during training, convergence may not be guaranteed.

The above two unique properties not only bring challenges to algorithm design but also make theoretical analysis much harder. There have been many efforts developing convergence guarantees for FL algorithms based on the assumptions of convexity and smoothness for the objective functions (Yu et al., 2019; Li et al., 2020c; Khaled et al., 2020). Although a recent study (Li et al., 2021) shows theoretical studies of FL on neural networks, its framework fails to generate multiple-local updates analysis, which is a key feature in FL. One of the most conspicuous questions to ask is:

Can we build a unified and generalizable convergence analysis framework for ReLU neural networks in FL?

In this paper, we give an affirmative answer to this question.

[1]Peking University, Beijing, China [2]The University of British Colombia, Vancouver, BC, Canada [3]Institute for Advanced Study, Princeton, NJ, United States [4]The University of Washington, Seattle, WA, United States. Correspondence to: Baihe Huang <baihehuang@pku.edu.cn>, Xiaoxiao Li <xiaoxiao.li@aya.yale.edu>, Zhao Song <magic.linuxkde@gmail.com>, Xin Yang <yx1992@cs.washington.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

It was recently proved in a series of papers that gradient descent converges to global optimum if the neural networks are overparameterized (significantly larger than the size of the datasets). Our work is motivated by Neural Tangent Kernel (NTK) that is originally proposed by (Jacot et al., 2018) and has been extensively studied over the last few years.

NTK is defined as the inner product space of pairwise data point gradient (aka Gram matrix). It describes the evolution of deep artificial neural networks during their training by gradient descent. Thus, we propose a novel NTK-based framework for federated learning convergence and generalization analysis on ReLU neural networks, so-called Federated Learning Neural Tangent Kernel (FL-NTK). Unlike the good property of the symmetric Gram matrix in classical NTK, we show the Gram matrix of FL-NTK is asymmetric in Section 3.2. Our techniques address the core question:

How shall we handle the asymmetric Gram matrix in FL-NTK?

Contributions. Our contributions are summarized into the following two folds:

- We proposed a framework to analyze federated learning in neural networks. By appealing to recent advances of over-parameterized neural networks, we prove convergence and generalization results of federated learning without the assumptions on the convexity of objective functions or distribution of data in the convergence analysis. Thus, we make the first step toward bridging the gap between the empirical success of federated learning and its theoretical understanding in the settings of ReLU neural networks. The results theoretically show that given fixed training instances, the number of communication rounds increases as the number of clients increases, which is also supported by empirical evidence. We show that when the neural networks are sufficiently wide, the training loss across all clients converges to zero at a linear rate. Furthermore, we also prove a data-dependent generalization bound.
- In federated learning, the update in the global model is no longer determined by the gradient directions directly. Indeed, gradients' heterogeneity in multiple local steps hinders the usage of standard neural tangent kernel analysis, which is based on the kernel gradient descent in the function space for a positive semi-definite kernel. We identify the dynamics of training loss by considering all intermediate states of local steps and establishing the tangent kernel space associated with a general non-symmetric Gram matrix to address this issue. We prove that this Gram matrix is close to symmetric at initialization using concentration properties at initialization. Therefore, we guarantee linear convergence results. This technique may further improve our understanding of many different FL optimization and aggregation methods on neural networks.

Organization. In Section 2 we discuss related work. In Section 3 we formulate FL convergence problem. In Section 4 we state our result. In Section 5 we summarize our technique overviews. In Section 6 we give a proof sketch of our result. In Section 7, we conduct experiments that affirmatively support our theoretical results. In Section 8 we conclude this paper and discuss future works. In Appendix A, we list several probability results. In Appendix B we prove our convergence result of FL-NTK. In Appendix C, we prove our generalization result of FL-NTK. The full version of this paper is available at `https://arxiv.org/abs/2105.05001`.

2. Related Work

Federated Learning Federated learning has emerged as an important paradigm in distributed deep learning. Generally, federated learning can be achieved by two approaches: 1) each party training the model using private data and where only model parameters being transferred and 2) using encryption techniques to allow safe communications between different parties (Yang et al., 2019a). In this way, the details of the data are not disclosed in between each party. In this paper, we focus on the first approach, which has been studied in (Dean et al., 2012; Shokri & Shmatikov, 2015; McMahan et al., 2016; 2017). Federated average (FedAvg) (McMahan et al., 2017) firstly addressed the communication efficiency problem by introducing a global model to aggregate local stochastic gradient descent updates. Later, different variations and adaptations have arisen. This encompasses a myriad of possible approaches, including developing better optimization algorithms (Li et al., 2020a; Wang et al., 2020) and generalizing model to heterogeneous clients under special assumptions (Zhao et al., 2018; Kairouz et al., 2021; Li et al., 2021).

Federated learning has been widely used in different fields. Healthcare applications have started to utilize FL for multi-center data learning to solve small data, and privacy in data sharing issues (Li et al., 2020b; Rieke et al., 2020; Li et al., 2019; Andreux et al., 2020). We have also seen new FL algorithms popping up (Wang et al., 2019; Lim et al., 2020; Chen et al., 2020a) in mobile edge computing. FL also has promising applications in autonomous driving (Liang et al., 2019), financial filed (Yang et al., 2019b), and so on.

Convergence of Federated Learning Despite its promising benefits, FL comes with new challenges to tackle, especially for its convergence analysis under communication-efficiency algorithms and data heterogeneity. The convergence of the general FL framework on neural networks is underexplored. A recent work (Li et al., 2021) studies FL convergence on one-layer neural networks. Nevertheless, it is limited by the assumption that each client performs a single local update epoch. Another line of approaches does not directly work on neural network setting (Li et al., 2020c; Khaled et al., 2020; Yu et al., 2019). Instead, they make assumptions on the convexity and smoothness of the objective functions, which are not realistic for non-linear neural networks.

Convergence of deep neural network The convergence of deep neural network (in non-FL setting) has been extensively studied over the last couple of years, Gaussian inputs (Zhong et al., 2017b; Li & Yuan, 2017; Zhong et al., 2017a; Ge et al., 2018; Zhong et al., 2019; Bakshi et al., 2019; Chen et al., 2020b), sufficiently wide neural network (Li & Liang, 2018; Du et al., 2019b; Allen-Zhu et al., 2019a;b; Du et al., 2019a; Zhang et al., 2020; Brand et al., 2021; Song & Zhang, 2021; Song et al., 2021), NTK (Jacot et al., 2018) and beyond NTK (Bai & Lee, 2019; Li et al., 2020d). For more literature, we refer the readers to textbook (Arora et al., 2020).

3. Problem Formulation as Neural Tangent Kernel

This section is organized as follows:

- We first introduce the notations used in our analysis.
- In Section 3.1, we present the basic setup of NTK and our federated learning algorithm.
- In Section 3.2, we give the analysis on the dynamics of NTK.

To capture the training dynamic of FL on ReLU neural networks, we formulate the problem in Neural Tangent Kernel regime.

Notations We use N to denote the number of clients and use c to denote its index. We use T to denote the number of communication rounds, and use t to denote its index. We use K to denote the number of local update steps, and we use k to denote its index. We use $u(t)$ to denote the aggregated server model after round t. We use $w_{k,c}(t)$ to denote c-th client's model in round t and step k.

Let $S_1 \cup S_2 \cup \cdots \cup S_N = [n]$ and $S_i \cap S_j = \emptyset$. Given n input data points and labels $\{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ in $\mathbb{R}^d \times \mathbb{R}$, the data of each client c is $\{(x_i, y_i) : i \in S_c\}$. Let $\phi(z) = \max\{z, 0\}$ denote the ReLU activation.

For each client $c \in [N]$, we use $y_c \in \mathbb{R}^{|S_c|}$ to denote the ground truth with regard to its data, and denote $y_c^{(k)}(t) \in \mathbb{R}^{|S_c|}$ to be the (local) model's output of its data in the t-th global round and k-th local step. For simplicity, we also use $y^{(k)}(t) \in \mathbb{R}^n$ to denote aggregating all (local) model's outputs in the t-th global round and k-th local step.

3.1. Preliminaries

In this subsection we introduce Algorithm 1, the brief algorithm of our federated learning (under NTK setting):

- In the t-th global round, server broadcasts $u(t) \in \mathbb{R}^{d \times m}$ to every client.
- Each client c then starts from $w_{0,c}(t) = u(t)$ and takes K (local) steps gradient descent to find $w_{K,c}(t)$.
- Each client sends $\Delta u_c(t) = w_{K,c}(t) - w_{0,c}(t)$ to server.
- Server computes a new $u(t+1)$ based on the average of all $\Delta u_c(t)$. Specifically, the server updates $u(t)$ by the average of all local updates $\Delta u_c(t)$ and arrives at
$$u(t+1) = u(t) + \eta_{\text{global}} \cdot \sum_{c \in [N]} \Delta u_c(t)/N.$$
- We repeat the above four steps by T times.

Setup We define one-hidden layer neural network function $f : \mathbb{R}^d \to \mathbb{R}$ similar as (Du et al., 2019b; Song & Yang, 2019; Brand et al., 2021; Song & Zhang, 2021)
$$f(u, x) := \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r \phi(u_r^\top x),$$
where $u \in \mathbb{R}^{d \times m}$ and $u_r \in \mathbb{R}^d$ is the r-th column of matrix u.

Definition 3.1 (Initialization)**.** *We initialize $u \in \mathbb{R}^{d \times m}$ and $a \in \mathbb{R}^m$ as follows:*

- *For each $r \in [m]$, u_r is sampled from $\mathcal{N}(0, \sigma^2 I)$.*
- *For each $r \in [m]$, a_r is sampled from $\{-1, +1\}$ uniformly at random (we don't need to train).*

We define the loss function for $j \in [N]$,
$$L_j(u, x) := \frac{1}{2} \sum_{i \in S_j} (f(u, x_i) - y_i)^2,$$
$$L(u, x) := \frac{1}{N} \sum_{j=1}^{N} L_j(u, x).$$

We can compute the gradient $\frac{\partial f(u,x)}{\partial u_r} \in \mathbb{R}^d$ (of function f),
$$\frac{\partial f(u, x)}{\partial u_r} = \frac{1}{\sqrt{m}} a_r x \mathbf{1}_{u_r^\top x \geq 0}.$$

Algorithm 1 Training Neural Network with FedAvg under NTK setting.

```
1:  $u_r(0) \sim \mathcal{N}(0, I_d)$ for $r \in [m]$.                                   ▷ $u \in \mathbb{R}^{d \times m}$
2:  for $t = 1, \dots, T$ do
3:      for $c = 1, \dots, N$ do
4:          /*Client receives weights from server*/
5:          $w_{0,c}(t) \leftarrow u(t)$                                                   ▷ $w_{0,c}(t), u(t) \in \mathbb{R}^{d \times m}$
6:          /*Client runs $K$ local steps gradient descent*/
7:          for $k = 1, \dots, K$ do
8:              for $i \in S_c$ do
9:                  $y_c^{(k)}(t)_i \leftarrow \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \phi(w_{k,c,r}(t)^\top x_i)$     ▷ $y_c^{(k)}(t) \in \mathbb{R}^{|S_c|}$
10:             end for
11:             for $r = 1 \to m$ do
12:                 for $i \in S_c$ do
13:                     $J_{i,:} \leftarrow \frac{1}{\sqrt{m}} a_r \phi'(w_{k,c,r}(t)^\top x_i) x_i^\top$     ▷ $J_{i,:} = \frac{\partial f(w_{k,c}(t), x_i, a)}{\partial w_{k,c,r}(t)} \in \mathbb{R}^{1 \times d}$
14:                 end for
15:                 $\mathrm{grad}_r \leftarrow -J^\top (y_c - y_c^{(k)}(t))$                    ▷ $J \in \mathbb{R}^{|S_c| \times d}$
16:                 $w_{k,c,r}(t) \leftarrow w_{k-1,c,r}(t) - \eta_{\mathrm{local}} \cdot \mathrm{grad}_r$
17:             end for
18:         end for
19:         /*Client sends weights to server*/
20:         $\Delta u_c \leftarrow w_{k,c}(t) - u(t)$
21:     end for
22:     /*Server aggregates the weights*/
23:     $\Delta u \leftarrow \frac{1}{N} \sum_{c \in [N]} \Delta u_c$                                ▷ $\Delta u \in \mathbb{R}^{d \times m}$
24:     $u(t+1) \leftarrow u(t) + \eta_{\mathrm{global}} \Delta u$                                ▷ $u(t+1) \in \mathbb{R}^{d \times m}$
25: end for
```

We can compute the gradient $\frac{\partial L_c(u)}{\partial u_r} \in \mathbb{R}^d$ (of function L),

$$\frac{\partial L_c(u)}{\partial u_r} = \frac{1}{\sqrt{m}} \sum_{i=1}^n (f(u, x_i) - y_i) a_r x_i \mathbf{1}_{u_{r,c}^\top x_i \geq 0}.$$

We formalize the problem as minimizing the sum of loss functions over all clients:

$$\min_{u \in \mathbb{R}^{d \times M}} L(u).$$

Local update In each local step, we update $w_{k,c}$ by gradient descent.

$$w_{k+1,c} \leftarrow w_{k,c} - \eta_{\mathrm{local}} \cdot \frac{\partial L_c(w_{k,c})}{\partial w_{k,c}}.$$

Note that

$$\frac{\partial L_c(w_{k,c})}{\partial w_{k,c,r}} = \frac{1}{\sqrt{m}} \sum_{i \in S_c} (f(w_{k,c}, x_i) - y_i) a_r x_i \mathbf{1}_{w_{k,c,r}^\top x_i \geq 0}.$$

Global aggregation In each global communication round we aggregate all local updates of clients by taking a simple average

$$\Delta u(t) = \sum_{c \in [N]} \Delta u_c(t)/N,$$

where $\Delta u_c(t) = w_{K,c} - w_{0,c}$ for all $c \in [N]$.

Global steps in total Then global model simply add $\Delta u(t)$ to its parameters.

$$u(t+1) \leftarrow u(t) + \eta_{\mathrm{global}} \cdot \Delta u(t).$$

3.2. NTK Analysis

The neural tangent kernel $H^\infty \in \mathbb{R}^{n \times n}$, introduced in (Jacot et al., 2018), is given by

$$H_{i,j}^\infty := \mathop{\mathbb{E}}_{w \sim \mathcal{N}(0,I)} \left[x_i^\top x_j \mathbf{1}_{w^\top x_i \geq 0, w^\top x_j \geq 0} \right]$$

At round t, let $y(t) = (y_1(t), y_2(t), \cdots, y_n(t)) \in \mathbb{R}^n$ be the prediction vector where $y_i(t) \in \mathbb{R}$ is defined as

$$y_i(t) = f(u(t), x_i).$$

Recall that we denote labels $y = (y_1, \cdots, y_n) \in \mathbb{R}^n$, $y_c = \{y_i\}_{i \in S_c}$, predictions $y(t) = (y(t)_1, \cdots, y(t)_n)$ and

$y_c(t) = \{y(t)_i\}_{i\in S_c}$, we can then rewrite $\|y - y(t)\|_2^2$ as follows:

$$\begin{aligned}&\|y - y(t+1)\|_2^2\\ =&\|y - y(t) - (y(t+1) - y(t))\|_2^2\\ =&\|y - y(t)\|_2^2 - 2(y - y(t))^\top(y(t+1) - y(t))\\ &+ \|y(t+1) - y(t)\|_2^2.\end{aligned} \tag{1}$$

Now we focus on $y(t+1) - y(t)$, notice for each $i \in [n]$

$$\begin{aligned}&y_i(t+1) - y_i(t)\\ =&\frac{1}{\sqrt{m}}\sum_{r=1}^m a_r(\phi(u_r(t+1)^\top x_i) - \phi(u_r^\top(t)x))\\ =&\frac{1}{\sqrt{m}}\sum_{r=1}^m a_r\Big(\phi\big((u_r(t) + \eta_{\text{global}}\Delta u_r(t))^\top x_i\big) -\\ &\phi(u_r^\top(t)x)\Big)\end{aligned} \tag{2}$$

where

$$\begin{aligned}\Delta u_r(t) :=\ &\frac{a_r}{N}\sum_{c\in[N]}\sum_{k\in[K]}\frac{\eta_{\text{local}}}{\sqrt{m}}\sum_{j\in S_c}\\ &(y_j - y_c^{(k)}(t)_j)x_j\mathbf{1}_{w_{k,c,r}(t)^\top x_j\ge 0}.\end{aligned}$$

In order to further analyze Eq (2), we separate neurons into two sets. One set contains neurons with the activation pattern changing over time and another set contains neurons with activation pattern holding the same. Specifically for each $i \in [n]$, we define the set $Q_i \subset [m]$ of neurons whose activation pattern is certified to hold the same throughout the algorithm

$$\begin{aligned}Q_i := \{r \in [m] : \forall w \in \mathbb{R}^d \text{ s.t. } \|w - w_r(0)\|_2 \le R,\\ \mathbf{1}_{w_r(0)^\top x_i\ge 0} = \mathbf{1}_{w^\top x_i\ge 0}\},\end{aligned}$$

and use $\overline{Q}_i$ to denote its complement. Then $y_i(t+1) - y_i(t) = v_{1,i} + v_{2,i}$ where

$$\begin{aligned}v_{1,i} =& \frac{1}{\sqrt{m}}\sum_{r\in Q_i} a_r\Big(\phi\big((u_r(t) + \eta_{\text{global}}\Delta u_r(t))^\top x_i\big)\\ &- \phi(u_r(t)^\top x_i)\Big),\\ v_{2,i} =& \frac{1}{\sqrt{m}}\sum_{r\in \overline{Q}_i} a_r\Big(\phi\big((u_r(t) + \eta_{\text{global}}\Delta u_r(t))^\top x_i\big)\\ &- \phi(u_r(t)^\top x_i)\Big).\end{aligned} \tag{3}$$

The benefit of this procedure is that v_1 can be written in closed form

$$v_{1,i} = \frac{\eta_{\text{global}}\eta_{\text{local}}}{Nm}\sum_{k\in[K],c\in[N]}\sum_{j\in S_c}\sum_{r\in Q_i} q_{k,c,j,r},$$

where

$$q_{k,c,j,r} := -(y_c^{(k)}(t)_j - y_j)x_i^\top x_j\mathbf{1}_{w_{k,c,r}(t)^\top x_j, u_r(t)^\top x_i\ge 0},$$

and v_2 is sufficiently small which we will show later.

Now, we extend the NTK analysis to FL. We start with defining the Gram matrix for f as follows.

Definition 3.2. *For any $t \in [0, T], k \in [K], c \in [N]$, we define matrix $H(t, k, c) \in \mathbb{R}^{n\times n}$ as follows*

$$\begin{aligned}H(t,k,c)_{i,j} &= \frac{1}{m}\sum_{r=1}^m x_i^\top x_j\mathbf{1}_{u_r^\top x_i\ge 0, w_{k,c,r}(t)^\top x_j\ge 0},\\ H(t,k,c)^\perp_{i,j} &= \frac{1}{m}\sum_{r\in\overline{Q}_i} x_i^\top x_j\mathbf{1}_{u_r^\top x_i\ge 0, w_{k,c,r}(t)^\top x_j\ge 0}.\end{aligned}$$

This Gram matrix is crucial for the analysis of error dynamics. When $t = 0$ and the width m approaches infinity, the H matrix becomes the NTK, and with infinite width, neural networks just behave like kernel methods with respect to the NTK (Arora et al., 2019b; Lee et al., 2020). It turns out that in the finite width case (Li & Liang, 2018; Allen-Zhu et al., 2019a;b; Du et al., 2019b;a; Song & Yang, 2019; Oymak & Soltanolkotabi, 2020; Huang & Yau, 2020; Chen & Xu, 2020; Zhang et al., 2020; Brand et al., 2021; Song & Zhang, 2021), the spectral property of the gram matrix also governs convergence guarantees for neural networks.

We can then decompose $-2(y - y(t))^\top(y(t+1) - y(t))$ into

$$\begin{aligned}&-2(y - y(t))^\top(y(t+1) - y(t))\\ =&-2(y - y(t))^\top(v_1 + v_2)\\ =&-\frac{2\eta_{\text{global}}\eta_{\text{local}}}{N}\sum_{i\in[n]}\sum_{k\in[K]}\sum_{c\in[N]}\sum_{j\in S_c} p_{i,k,c,j}\\ &-2\sum_{i\in[n]}(y_i - y_i(t))v_{2,i}\end{aligned} \tag{4}$$

where

$$\begin{aligned}p_{i,k,c,j} := &(y_i - y_i(t))\cdot(y_j - y_c^{(k)}(t)_j)\\ &\cdot(H(t,k,c)_{i,j} - H(t,k,c)^\perp_{i,j}).\end{aligned}$$

Now it remains to analyze Eq (1) and Eq (4). Our analysis leverages several key observations in the classical Neural Tangent Kernel theory throughout the learning process:

- Weights change lazily, namely
$$\|u(t+1) - u(t)\|_2 \le O(1/n).$$
- Activation patterns remain roughly the same, namely
$$\|H(t,k,c)^\perp\|_F \le O(1),$$

and

$$\|v_2\|_2 \le O(\|y - y(t)\|_2).$$

- Error controls model updates, namely

$$\|y(t+1) - y(t)\|_2 \le O(\|y - y(t)\|_2).$$

Based on the above observations, we show that the dynamics of federated learning is dominated in the following way

$$\begin{aligned} & \|y - y(t+1)\|_2^2 \\ \approx & \|y - y(t)\|_2^2 \\ & - 2 \sum_{k \in [K]} (y - y(t))^\top H(t,k)(y - y^{(k)}(t)), \end{aligned}$$

where the Gram matrix $H(t,k) \in \mathbb{R}^{n \times n}$ comes from combining the S_c columns of $H(t,k,c)$ for all $c \in [N]$. Since $S_1 \cup S_2 \cup \cdots \cup S_N = [n]$ and $S_i \cap S_j = \emptyset$, every $j \in [n]$ belongs to one unique S_c for some c and $H(t,k)_{i,j} = H(t,k,c)_{i,j}, j \in S_c$.

There are two difficulties lies in the analysis of these dynamics. First, unlike the symmetric Gram matrix in the standard NTK theory for centralized training, our FL framework's Gram matrix is asymmetric. Secondly, model update in each global round is influenced by all intermediate model states of all the clients.

To address these difficulties, we bring in two new techniques to facilitate our understanding of the learning dynamics.

- First, we generalize Theorem 4.2 in (Du et al., 2019b) to non-symmetric Gram matrices. We show in Lemma B.11 that with good initialization $H(t,k)$ is close to the original Gram matrix $H(0)$, so that model could benefit from a linear learning rate determined by the smallest eigenvalue of $H(0)$.

- Secondly, we leverage concentration properties at initialization to bound the difference between the errors in local steps and the errors in the global step. Specifically, we can show that $y - y^{(k)}(t) \approx y - y(t)$ for all $k \in [K]$.

4. Our Results

We first present the main result on the convergence of federated learning in neural networks by the following theorem.

Theorem 4.1 (Informal version of Theorem B.3). *Let*

$$m = \Omega(\lambda^{-4} n^4 \log(n/\delta)),$$

we iid initialize $u_r(0)$, a_r as Definition 3.1 where $\sigma = 1$. Let λ denote $\lambda_{\min}(H(0))$. Let κ denote the condition number of $H(0)$. For N clients, for any ϵ, let

$$T = O\Big(\frac{N}{\lambda \eta_{\text{local}} \eta_{\text{global}} K} \cdot \log(1/\epsilon)\Big),$$

there is an algorithm (FL-NTK) runs in T global steps and each client runs K local steps with choosing

$$\eta_{\text{local}} = O\Big(\frac{\lambda}{\kappa K n^2}\Big) \quad \text{and} \quad \eta_{\text{global}} = O(1)$$

and outputs weight U with probability at least $1-\delta$ such that the training loss $L(u,x)$ satisfies

$$L(u,x) = \frac{1}{2N} \sum_{i=1}^{N} (f(u,x_i) - y_i)^2 \le \epsilon.$$

We note that our theoretical framework is very powerful. With additional assumptions on the training data distribution and test data distribution, we can also show an upper bound for the generalization error of federated learning in neural networks. We first introduce a distributional assumption, which is standard in the literature (e.g, see (Arora et al., 2019a; Song & Yang, 2019)).

Definition 4.2 (Non-degenerate Data Distribution, Definition 5.1 in (Arora et al., 2019a)). *A distribution $\mathcal{D}$ over $\mathbb{R}^d \times \mathbb{R}$ is* (λ, δ, n)-non-degenerate*, if with probability at least $1-\delta$, for n i.i.d. samples $(x_i, y_i)_{i=1}^n$ chosen from $\mathcal{D}$, $\lambda_{\min}(H(0)) \ge \lambda > 0$.*

Our result on generalization bound is stated in the following theorem.

Theorem 4.3 (Informal, see Appendix C for details). *Fix failure probability $\delta \in (0,1)$. Suppose the training data $S = \{(x_i, y_i)\}_{i=1}^n$ are i.i.d samples from a $(\lambda, \delta/3, n)$-non-degenerate distribution $\mathcal{D}$, and*

- $\sigma \le O(\lambda \cdot \mathrm{poly}(\log n, \log(1/\delta))/n)$,
- $m \ge \Omega(\sigma^{-2} \cdot \mathrm{poly}(n, \log m, \log(1/\delta), \lambda^{-1}))$,
- $T \ge \Omega((\eta_{\text{local}} \eta_{\text{global}} K \lambda)^{-1} N \log(n/\delta))$.

We initialize $u \in \mathbb{R}^{d \times m}$ and $a \in \mathbb{R}^m$ as Definition 3.1. Consider any loss function $\ell : \mathbb{R} \times \mathbb{R} \to [0,1]$ that is 1-Lipschitz in its first argument. Then with probability at least $1-\delta$ the following event happens: after T global steps, the generalization loss

$$L_{\mathcal{D}}(f) := \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [\ell(f(u,x), y)]$$

is upper bounded as

$$L_{\mathcal{D}}(f) \le (2y^\top (H^\infty)^{-1} y/n)^{1/2} + O(\sqrt{\log(n/(\lambda\delta))/n}).$$

5. Techniques Overview

In our algorithm, when $K = 1$, i.e., we only perform one local step per global step, essentially we are performing gradient descent on all the n data points with step size $\eta_{\mathrm{global}}\eta_{\mathrm{local}}/N$. As the norm of the gradient is proportional to $1/\sqrt{m}$, when the neural network is sufficiently wide, we can control the norm of the gradient. Then by the update rule of gradient descent, we hence upper bound the movement of the first layer weight u. By anti-concentration of the normal distribution, this implies that for each input x, the activation status of most of the ReLU gates remains the same as initialized, which enables us to apply the standard convergence analysis of gradient descent on convex and smooth functions. Finally, we can handle the effect of ReLU gates whose activation status have changed by carefully choosing the step size.

However, the analysis becomes much more complicated when $K \geq 2$, where the movement of u is no longer determined by the gradient directly. Nevertheless, on each client, we are still performing gradient descent for K local steps. So we can handle the movement of the local weight w. The major technical bulk of this work is then proving that the training error shrinks when we set the global weight movement as the average of the local weight. Our argument is inspired by that of (Du et al., 2019b) but is much more involved.

6. Proof Sketch

In this section we sketch our proof of Theorem 4.1 and Theorem 4.3. The detailed proof is deferred to Appendix B.

In order to prove the linear convergence rate in Theorem 4.1, it is sufficient to show that the training loss shrinks in each round, or formally for each $\tau = 0, 1, \cdots$,

$$\|y(\tau+1) - y\|_2^2 \leq \Big(1 - \frac{\lambda \eta_{\mathrm{global}}\eta_{\mathrm{local}}K}{2N}\Big) \cdot \|y(\tau) - y\|_2^2.$$

We prove Eq. (5) by induction. Assume that we have proved for $\tau \leq t-1$ and we want to prove Eq. (5) for $\tau = t$. We first show that the movement of the weight u is bounded under the induction hypothesis.

Lemma 6.1 (Movement of global weight, informal version of Lemma B.12)**.** *For any $r \in [m]$,*

$$\|u_r(t) - u_r(0)\|_2 = O\Big(\frac{\sqrt{n}\|y - y(0)\|_2}{\sqrt{m}\lambda}\Big).$$

The detailed proof can be found in Appendix B.6.

We then turn to the proof of Eq. (5) by decomposing the loss in $t+1$-th global round

$$\begin{aligned} &\|y - y(t+1)\|_2^2 \\ &= \|y - y(t)\|_2^2 - 2(y - y(t))^\top (y(t+1) - y(t)) \\ &\quad + \|y(t+1) - y(t)\|_2^2 \end{aligned} \tag{5}$$

To this end, we need to investigate the change of prediction in consecutive rounds, which is described in Eq. (2). For the sake of simplicity, we introduce the notation

$$z_{i,r} := \phi\big((u_r(t) + \eta_{\mathrm{global}}\Delta u_r(t))^\top x_i\big) - \phi(u_r(t)^\top x_i),$$

then we have

$$\begin{aligned} y(t+1)_i - y(t)_i &= \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r z_{i,r} \\ &= \frac{1}{\sqrt{m}} \sum_{r \in Q_i} a_r z_{i,r} + v_{2,i}, \end{aligned}$$

where $v_{2,i}$ is introduced in Eq. (3).

For client $c \in [N]$ let $y_c^{(k)}(t)_j$ ($j \in S_c$) be defined by $y_c^{(k)}(t)_j = f(w_{k,c}(t), x_j)$. By the gradient-averaging scheme described in Algorithm 1, $\Delta u_r(t)$, the change in the global weights is

$$\frac{a_r}{N} \sum_{c \in [N]} \sum_{k \in [K]} \frac{\eta_{\mathrm{local}}}{\sqrt{m}} \sum_{j \in S_c} (y_j - y_c^{(k)}(t)_j) x_j \mathbf{1}_{w_{k,c,r}(t)^\top x_j \geq 0}.$$

Therefore, we can calculate $\frac{1}{\sqrt{m}} \sum_{r \in Q_i} a_r z_{i,r}$

$$\begin{aligned} &\frac{1}{\sqrt{m}} \sum_{r \in Q_i} a_r z_{i,r} \\ &= \frac{1}{\sqrt{m}} \sum_{r \in Q_i} a_r \Big(\phi\big((u_r(t) + \eta_{\mathrm{global}}\Delta u_r(t))^\top x_i\big) - \\ &\quad \phi(u_r(t)^\top x_i) \Big) \\ &= \frac{\eta_{\mathrm{global}}\eta_{\mathrm{local}}}{mN} \sum_{k \in [K]} \sum_{c \in [N]} \sum_{j \in S_c} \\ &\quad (y_j - y_c^{(k)}(t)_j) x_i^\top x_j \cdot \sum_{r \in Q_i} \mathbf{1}_{u_r^\top x_i \geq 0, w_{k,c,r}(t)^\top x_j \geq 0} \\ &= \frac{\eta_{\mathrm{global}}\eta_{\mathrm{local}}}{N} \sum_{k \in [K]} \sum_{c \in [N]} \sum_{j \in S_c} \\ &\quad (y_j - y_c^{(k)}(t)_j)(H(t,k,c)_{i,j} - H(t,k,c)_{i,j}^{\perp}). \end{aligned}$$

Further, we write $-2(y - y(t))^\top (y(t+1) - y(t))$ as follows:

$$\begin{aligned} &-2(y - y(t))^\top (y(t+1) - y(t)) \\ = &-2(y - y(t))^\top (v_1 + v_2) \\ = &-\frac{2\eta_{\mathrm{global}}\eta_{\mathrm{local}}}{N} \sum_{i \in [n]} \sum_{k \in [K]} \sum_{c \in [N]} \sum_{j \in S_c} \\ &(y_i - y_i(t))(y_j - y_c^{(k)}(t)_j)(H(t,k,c)_{i,j} - H(t,k,c)_{i,j}^{\perp}) \\ &-2 \sum_{i \in [n]} (y_i - y_i(t)) v_{2,i}. \end{aligned}$$

To summarize, we can decompose the loss as

$$\|y - y(t+1)\|_2^2 = \|y - y(t)\|_2^2 + C_1 + C_2 + C_3 + C_4.$$

where

$$\begin{aligned} C_1 = & -\frac{2\eta_{\text{global}}\eta_{\text{local}}}{N} \sum_{i\in[n]} \sum_{k\in[K]} \sum_{c\in[N]} \sum_{j\in S_c} \\ & (y_i - y_i(t))(y_j - y_c^{(k)}(t)_j) H(t,k,c)_{i,j}, \\ C_2 = & +\frac{2\eta_{\text{global}}\eta_{\text{local}}}{N} \sum_{i\in[n]} \sum_{k\in[K]} \sum_{c\in[N]} \sum_{j\in S_c} \\ & (y_i - y_i(t))(y_j - y_c^{(k)}(t)_j) H(t,k,c)_{i,j}^{\perp}, \\ C_3 = & -2 \sum_{i\in[n]} (y_i - y_i(t)) v_{2,i}, \\ C_4 = & +\|y(t+1) - y(t)\|_2^2. \end{aligned}$$

Let $R = \max_{r\in[m]} \|u_r(t) - u_r(0)\|_2$ be the maximal movement of global weights. Note that by Lemma 6.1, R can be made arbitrarily small as long as the width m is sufficiently large. Next, we bound C_1, C_2, C_3, C_4, by arguing that they are bounded from above if R is small and the learning rate is properly chosen. The detailed proof is deferred to Appendix B.3.

Lemma 6.2 (Bounding of C_1, C_2, C_3, C_4, informal version of Claim B.4, Claim B.5, Claim B.6 and Claim B.7)**.** *Assume that*

- $R = O(\lambda/n)$,
- $\eta_{\text{local}} = O(1/(\kappa K n^2))$,
- $\eta_{\text{global}} = O(1)$.

Then with high probability we have

$$\begin{aligned} C_1 \leq & -\eta_{\text{global}}\eta_{\text{local}}\lambda K \|y - y(t)\|_2^2/N, \\ C_2 \leq & +\eta_{\text{global}}\eta_{\text{local}}\lambda K \|y - y(t)\|_2^2/(40N), \\ C_3 \leq & +\eta_{\text{global}}\eta_{\text{local}}\lambda K \|y - y(t)\|_2^2/(40N), \\ C_4 \leq & +\eta_{\text{global}}\eta_{\text{local}}\lambda K \|y - y(t)\|_2^2/(40N). \end{aligned}$$

Combining the above lemma, we arrive to

$$\begin{aligned} & \|y - y(t+1)\|_2^2 \\ \leq & \left(1 - \frac{\eta_{\text{global}}\eta_{\text{local}}\lambda K}{2N}\right) \cdot \|y - y(t)\|_2^2, \end{aligned}$$

which completes the proof of Eq. (5). Finally Theorem 4.1 follows from

$$\begin{aligned} \|y - y(T)\|_2^2 \leq & \left(1 - \frac{\eta_{\text{global}}\eta_{\text{local}}\lambda K}{2N}\right)^T \\ \leq & \epsilon. \end{aligned}$$

7. Experiment

Models and datasets We examine our theoretical results on a benchmark dataset - Cifar10 in FL study. We perform 10 class classification tasks using ResNet56 (He et al., 2016). For fair convergence comparison, *we fixed the total number of samples* n. Based on our main result Theorem 4.1, we show the convergence with respect to the number of client N. To clearly evaluate the effects on N, we set all the clients to be activated (sampling all the clients) at each aggregation. We examine the settings of both non-iid and iid clients:

iid Data distribution is homogeneous in all the clients. Specifically, the label distribution over 10 classes is a uniform distribution.

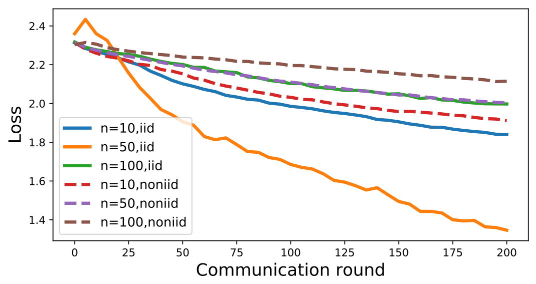
non-iid Data distribution is heterogeneous in all the clients. For non-iid splits, we utilize the Dirichlet distribution as in (Yurochkin et al., 2019; Wang et al., 2020; He et al., 2020a). First, each of these datasets is heterogeneously divided into J batches. The heterogeneous partition allocates $p_z \sim \text{Dir}_\text{J}(\alpha)$ proportion of the instances of label z to batch j. Then one label is sampled based on these vectors for each device, and an image is sampled without replacement based on the label. For all the classes, we repeat this process until all data points are assigned to devices.

Setup The experiment is conducted on one Ti2080 Nvidia GPU. We use SGD the the local optimizer with a learning rate of 0.03 to train the neural network.[1]. We set batch size as 128. We set the local update epoch $K = 1, 2, 5$ and 10[2]. We set Dirichlet distribution parameter for non-iid data as $\alpha = 0.5$. For all local update epoch settings, we set the client and sever communication round as 200. There are 50,000 training instances in Cifar10. We vary the number of clients $N = 10, 50, 100$ and record the training loss over communication rounds. The implementation is based on FedML (He et al., 2020b).

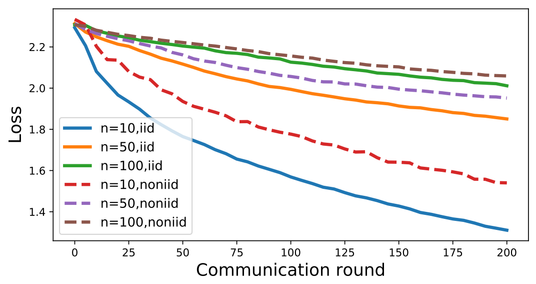
Impact of N Our theory suggests that given fixed training instances (fixed n) a smaller N requires less communication rounds to converge to a given ϵ. In other words, a smaller N may accelerate the convergence of the global model. Intuitively, a large N on a fixed amount of data means a heavier communication burden. Figure 1 shows the training loss curve of different choices of N. We empirically observe consistent results over different K that a smaller N converges faster given a fix number of total training samples, which is affirmative to our theoretical results.

[1]It is difficult to run real NTK experiment or full-batch gradient descent (GD) due to memory contrains.

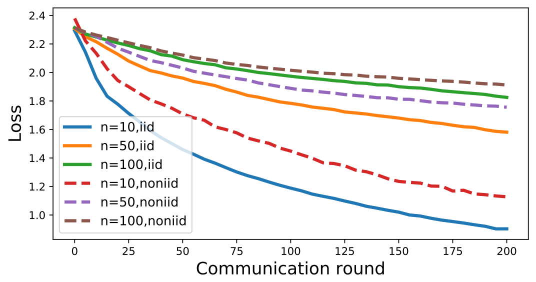
[2]Local update *step* equals to *epoch* in GD. But the number of *steps* of **one** *epoch* in SGD is equal to the number of mini-batches.



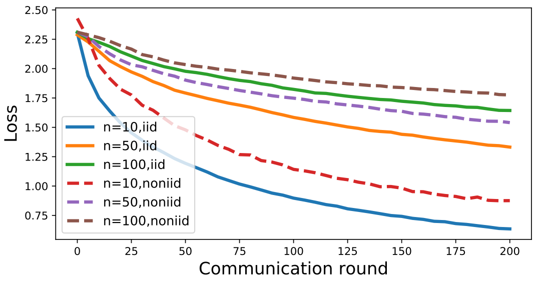
(a) Local update epoch K =1



(b) Local update epoch K =2



(c) Local update epoch K =5



(d) Local update epoch K =10

Figure 1: Training loss vs. communication rounds when number of clients $N = 10, 50, 100$ with iid and non-iid setting using mini-batch SGD optimizer.

8. Discussion

Overall, we provide the first comprehensive proof of convergence of gradient descent and generalization bound for over-parameterized ReLU neural network in federated learning. We consider the training dynamic with respect to local update steps K and number of clients N.

Different from most of existing theoretical analysis work on FL, FL-NTK prevails over the ability to exploit neural network parameters. There is a great potential for FL-NTK to understand the behavior of different neural network architectures, i.e., how Batch Normalization affects convergence in FL, like FedBN (Li et al., 2021). Different from FedBN, whose analysis is limited to $K = 1$, we provide the first general framework for FL convergence analysis by considering $K \geq 2$. The extension is non-trivial, as parameter update does not follow the gradient direction due to the heterogeneity of local data. To tackle this issue, we establish the training trajectory by considering all intermediate states and establishing an asymmetric Gram matrix related to local gradient aggregations. We show that with quartic network width, federated learning can converge to a global-optimal at a linear rate. We also provide a data-dependent generalization bound of over-parameterized neural networks trained with federated learning.

It will be interesting to extend the current FL-NTK framework for multi-layer cases. Existing NTK results of two-layer neural networks (NNs) have shown to be generalized to multi-layer NNs with various structures such as RNN, CNN, ResNet, etc. (Allen-Zhu et al., 2019a;b; Du et al., 2019a). The key techniques for analyzing FL on wide neural networks are addressed in our work. Our result can be generalized to multi-layer NNs by controlling the perturbation propagation through layers using well-developed tools (rank-one perturbation analysis, randomness decomposition, and extended McDiarmid's Inequality). We hope our results and techniques will provide insights for further study of distributed learning and other optimization algorithms.

Acknowledgements

Baihe Huang is supported by the Elite Undergraduate Training Program of School of Mathematical Sciences at Peking University, and work done while interning at Princeton University and Institute for Advanced Study (advised by Zhao Song). Xin Yang is supported by NSF grant CCF-2006359. This project is partially done while Xin was a Ph.D. student at University of Washington. The authors would like to thank the ICML reviewers for their valuable comments.

The full version of this paper is available at `https://arxiv.org/abs/2105.05001`.

References

Act, A. Health insurance portability and accountability act of 1996. *Public law*, 104:191, 1996.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning (ICML)*, pp. 242–252, 2019a.

Allen-Zhu, Z., Li, Y., and Song, Z. On the convergence rate of training recurrent neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019b.

Andreux, M., du Terrail, J. O., Beguier, C., and Tramel, E. W. Siloed federated learning for multi-centric histopathology datasets. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*, pp. 129–139. Springer, 2020.

Arora, R., Arora, S., Bruna, J., Cohen, N., Du, S., Ge, R., Gunasekar, S., Jin, C., Lee, J., Ma, T., Neyshabur, B., and Song, Z. Theory of deep learning. In *manuscript*, 2020.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, pp. 322–332, 2019a.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., and Wang, R. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8141–8150, 2019b.

Bai, Y. and Lee, J. D. Beyond linearization: On quadratic and higher-order approximation of wide neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

Bakshi, A., Jayaram, R., and Woodruff, D. P. Learning two layer rectified neural networks in polynomial time. In *Conference on Learning Theory (COLT)*, pp. 195–268. PMLR, 2019.

Bernstein, S. On a modification of chebyshev's inequality and of the error formula of laplace. *Ann. Sci. Inst. Sav. Ukraine, Sect. Math*, 1(4):38–49, 1924.

Brand, J. v. d., Peng, B., Song, Z., and Weinstein, O. Training (overparametrized) neural networks in near-linear time. In *Innovations in Theoretical Computer Science (ITCS)*, 2021.

Chen, L. and Xu, S. Deep neural tangent kernel and laplace kernel have the same rkhs. *arXiv preprint arXiv:2009.10683*, 2020.

Chen, M., Yang, Z., Saad, W., Yin, C., Poor, H. V., and Cui, S. A joint learning and communications framework for federated learning over wireless networks. *IEEE Transactions on Wireless Communications*, 2020a.

Chen, S., Klivans, A. R., and Meka, R. Learning deep relu networks is fixed-parameter tractable. *arXiv preprint arXiv:2009.13512*, 2020b.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., et al. Large scale distributed deep networks. In *Advances in neural information processing systems (NeurIPS)*, pp. 1223–1231, 2012.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning (ICML)*, pp. 1675–1685. PMLR, 2019a.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/pdf/1810.02054, 2019b.

Ge, R., Lee, J. D., and Ma, T. Learning one-hidden-layer neural networks with landscape design. In *International Conference on Learning Representations (ICLR)*, 2018.

He, C., Annavaram, M., and Avestimehr, S. Group knowledge transfer: Federated learning of large cnns at the edge. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020a.

He, C., Li, S., So, J., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., Shen, L., et al. Fedml: A research library and benchmark for federated machine learning. *NeurIPS Federated Learning workshop*, 2020b.

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

Huang, J. and Yau, H.-T. Dynamics of deep neural networks and neural tangent hierarchy. In *International Conference on Machine Learning (ICML)*, pp. 4542–4551, 2020.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems (NeurIPS)*, pp. 8571–8580, 2018.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 2021.

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local SGD on indentical and heterogeneous data. In *Proceedings of Artificial Intelligence and Statistics (AISTATS)*, 2020.

Lee, J. D., Shen, R., Song, Z., Wang, M., and Yu, Z. Generalized leverage score sampling for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Legislature, C. S. California consumer privacy act (ccpa). https://oag.ca.gov/privacy/ccpa, 2018.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. In *Conference on Machine Learning and Systems (MLSys)*, 2020a.

Li, W., Milletarì, F., Xu, D., Rieke, N., Hancox, J., Zhu, W., Baust, M., Cheng, Y., Ourselin, S., Cardoso, M. J., et al. Privacy-preserving federated brain tumour segmentation. In *International Workshop on Machine Learning in Medical Imaging*, pp. 133–141. Springer, 2019.

Li, X., Gu, Y., Dvornek, N., Staib, L., Ventola, P., and Duncan, J. S. Multi-site fmri analysis using privacy-preserving federated learning and domain adaptation: Abide results. *Medical Image Analysis*, 2020b.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. *International Conference on Learning Representations (ICLR)*, 2020c.

Li, X., Jiang, M., Zhang, X., Kamp, M., and Dou, Q. Fedbn: Federated learning on non-iid features via local batch normalization. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=6YEQUn0QICG.

Li, Y. and Liang, Y. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8157–8166, 2018.

Li, Y. and Yuan, Y. Convergence analysis of two-layer neural networks with ReLU activation. In *Advances in neural information processing systems (NIPS)*, pp. 597–607, 2017.

Li, Y., Ma, T., and Zhang, H. R. Learning over-parametrized two-layer neural networks beyond ntk. In *Conference on Learning Theory (COLT)*, pp. 2613–2682, 2020d.

Liang, X., Liu, Y., Chen, T., Liu, M., and Yang, Q. Federated transfer reinforcement learning for autonomous driving. *arXiv preprint arXiv:1910.06001*, 2019.

Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y.-C., Yang, Q., Niyato, D., and Miao, C. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3): 2031–2063, 2020.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282. PMLR, 2017.

McMahan, H., Moore, E., Ramage, D., and Agüera y Arcas, B. Federated learning of deep networks using model averaging. 2016.

Oymak, S. and Soltanolkotabi, M. Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. In *arXiv preprint*. https://arxiv.org/pdf/1902.04674.pdf, 2020.

Rieke, N., Hancox, J., Li, W., Milletari, F., Roth, H. R., Albarqouni, S., Bakas, S., Galtier, M. N., Landman, B. A., Maier-Hein, K., et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):1–7, 2020.

Shokri, R. and Shmatikov, V. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (CCS)*, pp. 1310–1321. ACM, 2015.

Song, Z. and Yang, X. Quadratic suffices for over-parametrization via matrix chernoff bound. *arXiv preprint arXiv:1906.03593*, 2019.

Song, Z. and Zhang, R. Hybrid quantum-classical implementation of training over-parameterized neural networks. In *manuscript*, 2021.

Song, Z., Yang, S., and Zhang, R. Does preprocessing help training over-parameterized neural networks. In *manuscript*, 2021.

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

Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6): 1205–1221, 2019.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):12, 2019a.

Yang, W., Zhang, Y., Ye, K., Li, L., and Xu, C.-Z. Ffd: a federated learning based method for credit card fraud detection. In *International Conference on Big Data*, pp. 18–32. Springer, 2019b.

Yu, H., Yang, S., and Zhu, S. 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 (AAAI)*, pp. 5693–5700, 2019.

Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., and Khazaeni, Y. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning (ICML)*, pp. 7252–7261, 2019.

Zhang, Y., Plevrakis, O., Du, S. S., Li, X., Song, Z., and Arora, S. Over-parameterized adversarial training: An analysis overcoming the curse of dimensionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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

Zhong, K., Song, Z., and Dhillon, I. S. Learning non-overlapping convolutional neural networks with multiple kernels. *arXiv preprint arXiv:1711.03440*, 2017a.

Zhong, K., Song, Z., Jain, P., Bartlett, P. L., and Dhillon, I. S. Recovery guarantees for one-hidden-layer neural networks. In *International Conference on Machine Learning (ICML)*, 2017b.

Zhong, K., Song, Z., Jain, P., and Dhillon, I. S. Provable non-linear inductive matrix completion. *Advances in Neural Information Processing Systems (NeurIPS)*, 32: 11439–11449, 2019.