Federated Adversarial Learning: A Framework with Convergence Analysis

Xiaoxiao Li [1] **Zhao Song** [2] **Jiaming Yang** [3]

Abstract

Federated learning (FL) is a trending training paradigm to utilize decentralized training data. FL allows clients to update model parameters locally for several epochs, then share them to a global model for aggregation. This training paradigm with multi-local step updating before aggregation exposes unique vulnerabilities to adversarial attacks. Adversarial training is a popular and effective method to improve the robustness of networks against adversaries. In this work, we formulate a *general* form of federated adversarial learning (FAL) that is adapted from adversarial learning in the centralized setting. On the client side of FL training, FAL has an inner loop to generate adversarial samples for adversarial training and an outer loop to update local model parameters. On the server side, FAL aggregates local model updates and broadcast the aggregated model. We design a global robust training loss and formulate FAL training as a min-max optimization problem. Unlike the convergence analysis in classical centralized training that relies on the gradient direction, it is significantly harder to analyze the convergence in FAL for three reasons: 1) the complexity of min-max optimization, 2) model not updating in the gradient direction due to the multi-local updates on the client-side before aggregation and 3) inter-client heterogeneity. We address these challenges by using appropriate gradient approximation and coupling techniques and present the convergence analysis in the over-parameterized regime. Our main result theoretically shows that the minimum loss under our algorithm can converge to ϵ small with chosen learning rate and communication rounds. It is noteworthy that our analysis is feasible for non-IID clients.

[1]University of British Columbia, BC, Canada. [2]Adobe Research, CA, USA. [3]University of Michigan, Ann Arbor, MI, USA. . Correspondence to: Zhao Song <zsong@adobe.com>, Jiaming Yang <jiamyang@umich.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Federated learning (FL) is playing an important role nowadays, as it allows different clients to train models collaboratively without sharing private information. One popular FL paradigm called FedAvg (McMahan et al., 2017) introduces an easy-to-implement distributed learning method without data sharing. Specifically, it requires a central server to aggregate model updates computed by the local clients (also known as nodes or participants) using local imparticipable private data. Then with these updates aggregated, the central server use them to train a global model.

Nowadays deep learning model are exposed to severe threats of adversarial samples. Namely, small adversarial perturbations on the inputs will dramatically change the outputs or output wrong answers (Szegedy et al., 2013). In this regard, much effort has been made to improve neural networks' resistance to such perturbations using adversarial learning (Tramèr et al., 2017; Samangouei et al., 2018; Madry et al., 2018). Among these studies, the adversarial training scheme in (Madry et al., 2018) has achieved the good robustness in practice. (Madry et al., 2018) proposes an adversarial training scheme that uses projected gradient descent (PGD) to generate alternative adversarial samples as the augmented training set. Generating adversarial examples during neural network training is considered as one of the most effective approaches for adversarial training up to now according to the literature (Carlini & Wagner, 2017; Athalye et al., 2018; Croce & Hein, 2020).

Although adversarial learning has attracted much attention in the centralized domain, its practice in FL is underexplored (Zizzo et al., 2020). Like training classical deep neural networks that use gradient-based methods, FL paradigms are vulnerable to adversarial samples. Adversarial learning in FL brings multiple open challenges due to FL properties on low convergence rate, application in non-IID environments, and secure aggregation solutions. Hence applying adversarial learning in an FL paradigm may lead to unstable training loss and a lack of robustness. However, a recent practical work (Zizzo et al., 2020) observed that although there exist difficulties of convergence, the feder-

ation of adversarial training with suitable hyperparameter settings can achieve adversarial robustness and acceptable performance. Motivated by the empirical results, we want to address the provable property of combining adversarial learning into FL from the theoretical perspective.

This work aims to theoretically study the unexplored convergence challenges that lie in the interaction between adversarial training and FL. To achieve a general understanding, we consider a general form of ***federated adversarial learning (FAL)***, which deploys adversarial training scheme on local clients in the most common FL paradigm, FedAvg (McMahan et al., 2017) system. Specifically, FAL has an inner loop of local updating that generates adversarial samples (i.e., using (Madry et al., 2018)) for adversarial training and an outer loop to update local model weights on the client side. Then global model is aggregated using FedAvg (McMahan et al., 2017). Our algorithm is detailed in Algorithm 1. We are interested in theoretically understanding the proposed FAL scheme from the aspects of model robustness and convergence:

Can FAL fit training data robustly and converge with an over-parameterized neural network?

The theoretical convergence analysis of adversarial training itself is challenging in the centralized training setting. (Tu et al., 2018) recently proposed a general theoretical method to analyze the risk bound with adversaries but did not address the convergence problem. The investigation of convergence on over-parameterized neural network has achieved tremendous progress (Du et al., 2019a; Allen-Zhu et al., 2019b;c; Du et al., 2019b; Arora et al., 2019b). The basic statement is that training can converge to sufficiently small training loss in polynomial iterations using gradient descent or stochastic gradient descent when the width of the network is polynomial in the number of training examples when initialized randomly. Recent theoretical analysis (Gao et al., 2019; Zhang et al., 2020b) extends these standard training convergence results to adversarial training settings. To answer the above interesting but challenging question, we formulate FAL as an min-max optimization problem. We extend the convergence analysis on the general formulation of over-parameterized neural networks in the FL setting that allows each client to perform min-max training and generate adversarial examples (see Algorithm 1). Involved challenges are arising in FL convergence analysis due to its unique optimization method: 1) unlike classical centralized setting, the global model of FL does not update in the gradient direction; 2) inter-client heterogeneity issue needs to be considered.

Despite the challenges, we give an affirmative answer to the above question. To the best of our knowledge, this work is the first theoretical study that studies those unexplored problems about the convergence of adversarial training with FL. The contributions of this paper are:

- We propose a framework to analyze a general form of FAL in over-parameterized neural networks. We follow a natural and valid assumption of data separability that the training dataset are well separated apropos of the adversarial perturbations' magnitude. After sufficient rounds of global communication and certain steps of local gradient descent for each t, we obtain the minimal loss close to zero. Notably, our assumptions do not rely on data distribution. Thus the proposed analysis framework is feasible for non-IID clients.

- We are the first to theoretically formulate the convergence of the FAL problem into a min-max optimization framework with the proposed loss function. In FL, the update in the global model is no longer directly determined by the gradient directions due to multiple local steps. To tackle the challenges, we define a new 'gradient', *FL gradient*. With valid ReLU Lipschitz and over-parameterized assumptions, we use gradient coupling for gradient updates in FL to show the model updates of each global updating is bounded in federated adversarial learning.

2. Related Work

Federated Learning. A efficient and privacy-preserving way to learn from the distributed data collected on the edge devices (a.k.a clients) would be FL. FedAvg is a easy-to-implement distributed learning strategy by aggregating local model updates of the server's side, and then transmitting the averaged model back to the local clients. Later, many FL methods are developed baed on FedAvg. Theses FL schemes can be divided into aggregation schemes (McMahan et al., 2017; Wang et al., 2020; Li et al., 2021) and optimization schemes (Reddi et al., 2020; Zhang et al., 2020a). Nearly all the them have the common characteristics that client model are updating using gradient descent-based methods, which is venerable to adversarial attacks. In addition, data heterogeneity brings in huge challeng in FL. For IID data, FL has been proven effective. However, in practice, data mostly distribute as non-IID. Non-IID data could substantially degrade the performance of FL models (Zhao et al., 2018; Li et al., 2019; 2021; 2020a). Despite the potential risk in security and unstable performance in non-IID setting, as FL mitigates the concern of data sharing, it is still a popular and practical solution for distributed data learning in many real applications such as healthcare (Li et al., 2020b; Rieke et al., 2020), autonomous driving (Liang et al., 2019), IoTs (Wang et al., 2019; Lim et al., 2020). Sketching techniques have also been applied to federated learning and shown provable guarantees (Song et al., 2023). (Bian et al., 2023) generalizes previous classical

newton method (in the area of robust central path for solving linear programming (Cohen et al., 2019; Song & Yu, 2021; Jiang et al., 2021), semi-definite programming (Jiang et al., 2020; Huang et al., 2022; Gu & Song, 2022) and empirical risk minimization (Lee et al., 2019; Qin et al., 2023)) to federated learning setting.

Learning with Adversaries. Since the discovery of adversarial examples (Szegedy et al., 2013), many effective defense methods have been proposed to make neural networks robust to perturbations. As adversarial examples are an issue of robustness, the popular scheme is to include learning with adversarial examples, which can be traced back to (Goodfellow et al., 2014). It produces adversarial examples and injecting them into training data. Later, Madry et al. (Madry et al., 2018) proposed training on multi-step PGD adversaries and empirically observed that adversarial training consistently achieves small and robust training loss in wide neural networks. (Gao et al., 2022) provides a sublinear time adversarial learning algorithm with provable guarantee under NTK regime.

Federated Adversarial Learning. The fact that adversarial examples are often misclassified poses potential security threats for practical machine learning applications. Adversarial training (Goodfellow et al., 2014; Kurakin et al., 2016) is a popular protocol to train robust models by inserting adversarial examples in training. The use of adversarial training in FL presents a number of open challenges, including poor convergence due to multiple local update steps, instability and heterogeneity of clients, cost and security request of communication, and so on. To defend the adversarial attacks in federated learning, limited recent studies have proposed to include adversarial training on clients in the local training steps (Bhagoji et al., 2019; Zizzo et al., 2020). These two works empirically showed the performance of adversarial training, while the theoretical analysis of convergence is under explored. (Deng et al., 2020) focused the problem of distributionally robust FL with an emphasis on reducing the communication rounds, they traded $O(T^{1/8})$ convergence rate for $O(T^{1/4})$ communication rounds. In addition, different from our focus on a generic theoretical analysis framework, (Zhang et al., 2021) is a methodology paper that proposed an adversarial training strategy in classical distributed setting, with focus on specific training strategy (PGD, FGSM), which could be generalized to a method in FAL.

Convergence via Over-parameterization. Convergence analysis on over-parameterized neural networks falls in two lines. In the first line of work (Li & Liang, 2018; Allen-Zhu et al., 2019b;c;a) data separability plays a crucial role, and is widely used in theoretically showing the convergence result in the over-parameterized neural network setting. To be specific, data separability theory shows that to guarantee convergence, the width (m) of a neural network shall be at least polynomial factor of all parameters (i.e. $m \geq \mathrm{poly}(n, d, 1/\delta)$), where δ is the minimum distance between all pairs of data points, n is the number of data points and d is the data dimension. Another line of work (Du et al., 2019b; Arora et al., 2019a;b; Song & Yang, 2019; Brand et al., 2021; Song et al., 2021a;b; Huang et al., 2021; Zhang, 2022; Munteanu et al., 2022; Hu et al., 2022; Alman et al., 2022; Zhang, 2022; Gao et al., 2023a) builds on neural tangent kernel (NTK) (Jacot et al., 2018). In this line of work, the minimum eigenvalue of the NTK is required to be lower bounded to guarantee convergence. Our analysis focuses on the former approach based on data separability.

Robustness of Federated Learning. Previously there were several works that theoretically analyzed the robustness of federated learning under noise. (Yin et al., 2018) developed distributed optimization algorithms that were provably robust against arbitrary and potentially adversarial behavior in distributed computing systems, and mainly focused on achieving optimal statistical performance. (Reisizadeh et al., 2020) developed a robust federated learning algorithm by considering a structured affine distribution shift in users' data. Their analysis was built on several assumptions on the loss functions without a direct connection to neural network.

3. Problem Formulation

To explore the properties of FAL in deep learning, we formulate the problem in over-parameterized neural network regime. We start by presenting the notations and setup required for federated adversarial learning, then we will describe the loss function we use and our FAL algorithm.

3.1. Notations

For a vector x, we use $\|x\|_p$ to denote its ℓ_p norm, in this paper we mainly consider the situation when $p = 1, 2$, or ∞. For a matrix $U \in \mathbb{R}^{d \times m}$, we use $U^\top$ to denote its transpose and use $\mathrm{tr}[U]$ to denote its trace. We use $\|U\|_1$ to denote its entry-wise ℓ_1 norm. We use $\|U\|_2$ to denote its spectral norm. We use $\|U\|_F$ to denote its Frobenius norm. For $j \in [m]$, we let $U_j \in \mathbb{R}^d$ be the j-th column of U. We let $\|U\|_{2,1}$ denotes $\sum_{j=1}^m \|U_j\|_2$. We let $\|U\|_{2,\infty}$ denotes $\max_{j \in [m]} \|U_j\|_2$.

We denote Gaussian distribution with mean μ and covariance Σ as $\mathcal{N}(\mu, \Sigma)$. We use $\sigma(\cdot)$ to denote the ReLU function $\sigma(x) = \max\{x, 0\}$, and use $\mathbb{1}\{A\}$ to denote the indicator function of event A.

3.2. Problem Setup

Two-Layer ReLU Network in FAL. Following recent theoretical work in understanding neural networks training in deep learning (Du et al., 2019b; Arora et al., 2019a;b; Song & Yang, 2019; Lee et al., 2020; Song et al., 2021a; Brand et al., 2021; Zhang, 2022; Alman et al., 2022; Hu et al., 2022), in this paper, we focus on a two-layer neural network that has m neurons in the hidden layer, where each neuron is a ReLU activation function. Formally we have the following definitions.

We define the global network as

$$f_U(x) := \sum_{r=1}^{m} a_r \cdot \sigma(\langle U_r, x\rangle + b_r) \tag{1}$$

and for $c \in [N]$, we define the local network of client c as

$$f_{W_c}(x) := \sum_{r=1}^{m} a_r \cdot \sigma(\langle W_{c,r}, x\rangle + b_r). \tag{2}$$

Here $U = (U_1, U_2, \ldots, U_m) \in \mathbb{R}^{d\times m}$ is the global hidden weight matrix, $W_c = (W_{c,1}, \ldots, W_{c,m}) \in \mathbb{R}^{d\times m}$ is the local hidden weight matrix of client c, and $a = (a_1, a_2, \ldots, a_m) \in \mathbb{R}^m$ denotes the output weight, $b = (b_1, b_2, \ldots, b_m) \in \mathbb{R}^m$ denotes the bias.

During the process of federated adversarial learning, we only update the value of U and W, while keeping a and b equal to their initialization, so we can write the global network as $f_U(x)$ and the local network as $f_{W_c}(x)$. For the situation we don't care about the weight matrix, we write $f(x)$ or $f_c(x)$ for short. Next, we make some standard assumptions regarding our training set.

Definition 3.1 (Dataset)**.** *There are N clients and $n = NJ$ data in total. Let $\mathcal{S} = \cup_{c\in[N]}\mathcal{S}_c$ where $\mathcal{S}_c = \{(x_{c,1}, y_{c,1}), ..., (x_{c,J}, y_{c,J})\} \subseteq \mathbb{R}^d \times \mathbb{R}$ denotes the J training data of client c. Without loss of generality, we assume $\|x_{c,j}\|_2 = 1$ holds for all $c \in [N], j \in [J]$, and the last coordinate of each point equals to $1/2$, so we consider $\mathcal{X} := \{x \in \mathbb{R}^d : \|x\|_2 = 1,\ x_d = 1/2\}$. For simplicity, we assume that $|y_{c,j}| \le 1$ holds for all $c \in [N]$ and $j \in [J]$.*[1]

We now define the initialization for the neural networks.

Definition 3.2 (Initialization)**.** *The initialization of $a \in \mathbb{R}^m, U \in \mathbb{R}^{d\times m}, b \in \mathbb{R}^m$ is $a(0) \in \mathbb{R}^m, U(0) \in \mathbb{R}^{d\times m}, b(0) \in \mathbb{R}^m$. The initialization of client c's local weight matrix W_c is $W_c(0,0) = U(0)$. Here the second term in W_c denotes iteration of local steps.*

- *For each $r \in [m]$, $a_r(0)$ are i.i.d. sampled from $[-1/m^{1/3}, +1/m^{1/3}]$ uniformly.*
- *For each $i \in [d], r \in [m]$, $U_{i,r}(0)$ and $b_r(0)$ are i.i.d. random Gaussians sampled from $\mathcal{N}(0, 1/m)$. Here $U_{i,r}$ means the (i,r)-entry of U.*

For each global iteration $t \in [T]$,

- *For each $c \in [N]$, the initial value of client c's local weight matrix W_c is $W_c(t,0) = U(t)$.*

Next we formulate the adversary model that will be used.

Definition 3.3 (ρ-Bounded adversary)**.** *Let $\mathcal{F}$ denote the function class. An adversary is a mapping $\mathcal{A} : \mathcal{F} \times \mathcal{X} \times \mathbb{R} \to \mathcal{X}$ which denotes the adversarial perturbation. For $\rho > 0$, we define the ℓ_2 ball $\mathcal{B}_2(x,\rho) := \{\widetilde{x} \in \mathbb{R}^d : \|\widetilde{x} - x\|_2 \le \rho\} \cap \mathcal{X}$, we say an adversary $\mathcal{A}$ is **ρ-bounded** if it satisfies $\mathcal{A}(f, x, y) \in \mathcal{B}_2(x,\rho)$.*

*Furthermore, given $\rho > 0$, we denote the **worst-case** adversary as $\mathcal{A}^* := \operatorname{argmax}_{\widetilde{x}\in\mathcal{B}_2(x,\rho)} \ell(f(\widetilde{x}), y)$, where ℓ is defined in Definition 3.5.*

Well-Separated Training Set. In the over-parameterized regime, it is standard to assume that the training set is well-separated. Since we deal with adversarial perturbations, we require the following γ-separability, which is a bit stronger.

Definition 3.4 (γ-separability)**.** *Let $\gamma \in (0, 1/2), \delta \in (0, 1/2), \rho \in (0, 1/2)$ denote three parameters such that $\gamma \le \delta \cdot (\delta - 2\rho)$. We say our training set*

$$\mathcal{S} = \cup_{c\in[N]}\mathcal{S}_c = \cup_{c\in[N], j\in[J]}\{(x_{c,j}, y_{c,j})\} \subset \mathbb{R}^d \times \mathbb{R}$$

*is **globally γ-separable** w.r.t a ρ-bounded adversary, if $\|x_{c_1,j_1} - x_{c_2,j_2}\|_2 \ge \delta$ holds for any $c_1 \ne c_2$ and $j_1 \ne j_2$.*

Note that in the above definition, the introducing of γ is for expression simplicity of Theorem 4.1, and the assumption $\gamma \le \delta \cdot (\delta - 2\rho)$ is reasonable and easy to achieve in adversarial training. It is also noteworthy that, our problem setup does not need the assumption on independent and identically distribution (IID) on data, thus such a formation can be applied to unique challenge of the non-IID setting in FL.

3.3. Federated Adversarial Learning

Adversary and Robust Loss. We set the following loss for sake of technical presentation simplicity, as is customary in prior studies (Gao et al., 2019; Allen-Zhu et al., 2019a):

Definition 3.5 (Lipschitz convex loss)**.** *A loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a **Lipschitz convex loss**, if it satisfies the following four properties:*

1. *non-negative;*

[1] Our assumptions on data points are reasonable since we can do scale-up. In addition, ℓ_2 norm normalization is a typical technique in experiments. Same assumptions appears in many previous works like (Arora et al., 2019b; Allen-Zhu et al., 2019a;b).

2. *convex w.r.t. the first input of ℓ;*
3. 1-*Lipshcitz, which means* $|\ell(x_1, y_1) - \ell(x_2, y_2)| \leq \|(x_1, y_1) - (x_2, y_2)\|_2$;
4. $\ell(y, y) = 0$ *for all* $y \in \mathbb{R}$.

We assume ℓ is a Lipschitz convex loss.[2] Next we define our robust loss function of a network, which is based on the adversarial samples generated by a ρ-bounded adversary $\mathcal{A}$.

Definition 3.6 (Training loss)**.** *Given a client's training set $\mathcal{S}_c = \{(x_{c,j}, y_{c,j})\}_{j=1}^J \subset \mathbb{R}^d \times \mathbb{R}$ of J samples. Let $f_c : \mathbb{R}^d \to \mathbb{R}$ be the neural network. We define the classical training loss of f_c as*

$$\mathcal{L}(f_c, S_c) := \frac{1}{J} \sum_{j=1}^{J} \ell\left(f_c(x_{c,j}), y_{c,j}\right).$$

Given $\mathcal{S} = \cup_{c\in[N]} \mathcal{S}_c$, we define the global loss as

$$\mathcal{L}(f_U, S) := \frac{1}{NJ} \sum_{c=1}^{N} \sum_{j=1}^{J} \ell(f_U(x_{c,j}), y_{c,j}).$$

Given an adversary $\mathcal{A}$ that is ρ-bounded, we define the global loss with respect to $\mathcal{A}$ as

$$\mathcal{L}_{\mathcal{A}}(f_U) := \frac{1}{NJ} \sum_{c=1}^{N} \sum_{j=1}^{J} \ell(f_U(\mathcal{A}(f_c, x_{c,j}, y_{c,j})), y_{c,j})$$

*and define global robust loss (in terms of **worst-case**) as*

$$\mathcal{L}_{\mathcal{A}^*}(f_U) := \frac{1}{NJ} \sum_{c=1}^{N} \sum_{j=1}^{J} \ell(f_U(\mathcal{A}^*(f_c, x_{c,j}, y_{c,j})), y_{c,j})$$

Moreover, since we deal with pseudo-net (Definition 5.1), we also define the local loss of a pseudo-net as

$$\mathcal{L}(g_c, \mathcal{S}_c) := \frac{1}{J} \sum_{j=1}^{J} \ell\left(g_c(x_{c,j}), y_{c,j}\right)$$

and the global loss of a pseudo-net as

$$\mathcal{L}(g_U, \mathcal{S}) := \frac{1}{NJ} \sum_{c=1}^{N} \sum_{j=1}^{J} \ell(g_U(x_{c,j}), y_{c,j}).$$

Algorithm. We focus on a general FAL framework that is adapted from the most common adversarial training in the classical setting on the client. Specifically, we describe the adversarial learning of a local neural network f_{W_c} against an adversary $\mathcal{A}$ that generate adversarial examples during training as shown in Algorithm 1. As for the analysis of a general theoretical analysis framework, we do not specify the explicit format of $\mathcal{A}$.

The FAL algorithm contains two procedures: one is CLIENTUPDATE running on client side and the other is SERVEREXECUTION running on server side. These two procedures are iteratively processed through communication iterations. Adversarial training is addressed in procedure CLIENTUPDATE. Hence, there are two loops in CLIENTUPDATE procedure: the outer loop is iteration for local model updating; and the inner loop is iteratively generating adversarial samples by the adversary $\mathcal{A}$. In the outer loop in SERVEREXECUTION procedure, the neural network's parameters are updated to reduce its prediction loss on the new adversarial samples.

4. Our Result

The main result of this work is showing the convergence of FAL algorithm (Algorithm 1) in overparameterized neural networks. Specifically, our defined global training loss (Definition 3.6) converges to a small ϵ with the chosen communication round T, local and global learning rate η_{local}, η_{global}. We now formally present our main result.

Theorem 4.1 (Federated Adversarial Learning)**.** *Let $c_0 \in (0, 1)$ be a fixed constant. Let N denotes the total number of clients and J denotes the number of data points per client. Suppose that our training set $\mathcal{S} = \cup_{c\in[N]} \mathcal{S}_c$ is globally γ-separable for some $\gamma > 0$. Then, for all $\epsilon \in (0, 1)$, there exists $R = \text{poly}((NJ/\epsilon)^{1/\gamma})$ that satisfies: for every $K \geq 1$ and $T \geq \text{poly}(R/\epsilon)$, for all $m \geq \text{poly}(d, (NJ/\epsilon)^{1/\gamma})$, with probability $\geq 1 - \exp(-\Omega(m^{1/3}))$, running **federated adversarial learning** (Algorithm 1) with step size choices*

$$\eta_{\text{global}} = 1/\text{poly}(NJ, R, 1/\epsilon) \quad \textit{and} \quad \eta_{\text{local}} = 1/K$$

will output a list of weights $\{U(1), U(2), \cdots, U(T)\} \in \mathbb{R}^{d\times m}$ that satisfy:

$$\min_{t\in[T]} \mathcal{L}_{\mathcal{A}}(f_{U(t)}) \leq \epsilon.$$

Discussion. In Theorem 4.1, one key element that affects parameters m, R, T is the data separability γ. As the data separability bound becomes larger, the parameter R becomes smaller, resulting in the need of a larger global learning rate η_{global} to achieve convergence. We conduct experiments in Section 6 to verify Theorem 4.1 empirically.

[2] Lipschitz is a common and reasonable in deep learning and large language models, since a number of work have proved certain Lipschitz properties for those learning models(Li et al., 2023b; Deng et al., 2023; Gao et al., 2023b; Li et al., 2023a).

Algorithm 1 Federated Adversarial Learning (FAL)

Notations: Training sets of clients with each client is indexed by c, $\mathcal{S}_c = \{(x_{c,j}, y_{c,j})\}_{j=1}^{J}$; adversary $\mathcal{A}$; local learning rate η_{local}; global learning rate η_{global}; local updating iterations K; global communication round T.

Initialization $a(0) \in \mathbb{R}^m, U(0) \in \mathbb{R}^{d\times m}, b(0) \in \mathbb{R}^m$
For $t = 0 \to T$, we iteratively run **Procedure A** then **Procedure B**
procedure A. CLIENTUPDATE(t, c)
  $\mathcal{S}_c(t) \leftarrow \emptyset$
  $W_c(t, 0) \leftarrow U(t)$ ▷ Receive global model weights update.
  for $k = 0 \to K - 1$ **do**
    for $j = 1 \to J$ **do**
      $\widetilde{x}_{c,j}^{(t)} \leftarrow \mathcal{A}(f_{W_c(t,k)}, x_{c,j}, y_{c,j})$ ▷ Adversarial samples. f_{W_c} is defined as (2).
      $\mathcal{S}_c(t) \leftarrow \mathcal{S}_c(t) \cup (\widetilde{x}_{c,j}^{(t)}, y_{c,j})$
    end for
    $W_c(t, k+1) \leftarrow W_c(t, k) - \eta_{\text{local}} \cdot \nabla_{W_c}\mathcal{L}(f_{W_c(t,k)}, \mathcal{S}_c(t))$
  end for
  $\Delta U_c(t) \leftarrow W_c(t, K) - U(t)$
  Send $\Delta U_c(t)$ to SERVEREXECUTION
end procedure
procedure B. SERVEREXECUTION(t):
  for each client c **in parallel do do**
    $\Delta U_c(t) \leftarrow$ CLIENTUPDATE(c, t) ▷ Receive local model weights update.
    $\Delta U(t) \leftarrow \frac{1}{N}\sum_{c\in[N]} \Delta U_c(t)$
    $U(t+1) \leftarrow U(t) + \eta_{\text{global}} \cdot \Delta U(t)$ ▷ Aggregation on the server side.
    Send $U(t+1)$ to client c for CLIENTUPDATE(c, t)
  end for
end procedure

5. Proof Sketch

To handle the min-max objective in FAL, we formulate the optimization of FAL in the framework of *online gradient descent*[3]: at each local step k on the client side, firstly the adversary generates adversarial samples and computes the loss function $\mathcal{L}\left(f_{W_c(t,k)}, \mathcal{S}_c(t)\right)$, then the local client learner takes the fresh loss function and update

$$W_c(t, k+1) = W_c(t, k) - \eta_{\text{local}} \cdot \nabla_{W_c}\mathcal{L}\left(f_{W_c(t,k)}, \mathcal{S}_c(t)\right).$$

Compared with the centralized setting, the key difficulties in the convergence analysis of FL are induced by multiple local step updates of the client side and the step updates on both local and global sides. Specifically, local updates are not the standard gradient as the centralized adversarial training when $K \geq 2$. We use $-\Delta U(t)$ in substitution of the real gradient of U to update the value of $U(t)$. This brings in challenges to bound the gradient of the neural networks. Nevertheless, gradient bounding is challenging in adversarial training solely. To this end, we use gradient coupling method twice to solve this core problem: firstly we bound the difference between real gradient and FL gradient (defined below), then we bound the difference between pseudo gradient and real gradient.

[3]We refer the readers to (Hazan, 2016) for more details regarding online gradient descent.

5.1. Existence of Small Robust Loss

In this section, we denote $U(t)$ as the global weights of communication round t. U^* is the value of U after small perturbations from $\widetilde{U}$ which satisfies $\|U^* - \widetilde{U}\|_{2,\infty} \leq R/m^{c_1}$, here $c_1 \in (0, 1)$ is a constant (e.g. $c_1 = 2/3$), m is the width of the neural network and R is a parameter. We will specify the value of these parameters in appendix.

We study the over-parameterized neural nets' well-approximated pseudo-network to learn gradient descent for over-parameterized neural nets whose weights are close to initialization. At a high level, pseudo-network can be seen as a linear approximation of our two layer ReLU neural network near initialization, and the introducing of pseudo-network makes the proof more intuitive due to its linearity.

Definition 5.1 (Pseudo-network)**.** *Given weights $U \in \mathbb{R}^{d\times m}$, $a \in \mathbb{R}^m$ and $b \in \mathbb{R}^m$, for a neural network $f_U(x) = \sum_{r=1}^{m} a_r \cdot \sigma(\langle U_r, x\rangle + b_r)$, we define the corresponding **pseudo-network** $g_U : \mathbb{R}^d \to \mathbb{R}$ as*

$$g_U(x) := \sum_{r=1}^{m} a_r \cdot \langle U_r(t) - U_r(0), x\rangle \cdot \mathbb{1}\{\langle U_r(0), x\rangle + b_r \geq 0\}.$$

To obtain our main Theorem 4.1, we first show in Theorem 5.2 that we can find a U^* which is close to $U(0)$ and also makes $\mathcal{L}_{\mathcal{A}^*}(f_{U^*})$ sufficiently small. Later in Theorem 5.6 we show that the average of $\mathcal{L}_{\mathcal{A}}(f_{U(t)})$ is dominated by $\mathcal{L}_{\mathcal{A}^*}(f_{U^*})$, thus we can prove the minimum of $\mathcal{L}_{\mathcal{A}}(f_{U(t)})$ is also ϵ small.

Theorem 5.2 (Existence result, informal version of Theorem F.3)**.** *For all $\epsilon \in (0,1)$, there exists*

$$M_0 = \mathrm{poly}(d, (NJ/\epsilon)^{1/\gamma}) \quad \textit{and} \quad R = \mathrm{poly}((NJ/\epsilon)^{1/\gamma})$$

satisfying: for all $m \geq M_0$, with probability at least $1 - \exp(-\Omega(m^{c_0}))$ there exists $U^* \in \mathbb{R}^{d\times m}$ such that

$$\|U^* - U(0)\|_{2,\infty} \leq R/m^{c_1} \quad \textit{and} \quad \mathcal{L}_{\mathcal{A}^*}(f_{U^*}) \leq \epsilon.$$

The full proof of Theorem 5.2 is in appendix.

5.2. Convergence Result for FAL

Definition 5.3 (Local gradient)**.** *For a local real network denoted by $f_{W_c(t,k)}$, we denote its gradient by*

$$\nabla(f_c, t, k) := \nabla_{W_c}\mathcal{L}(f_{W_c(t,k)}, \mathcal{S}_c(t)).$$

If the corresponding pseudo-network is $g_{W_c(t,k)}$, we denote its gradient by

$$\nabla(g_c, t, k) := \nabla_{W_c}\mathcal{L}(g_{W_c(t,k)}, \mathcal{S}_c(t)).$$

Now we consider the global neural network. We define **pseudo gradient** as $\nabla(g,t) := \nabla_U \mathcal{L}(g_{U(t)}, \mathcal{S}(t))$ and define **FL gradient** as $\widetilde{\nabla}(f,t) := -\frac{1}{N}\Delta U(t)$, which is used in the proof of Theorem 5.6. Next we present the following two lemmas. We call them **gradient coupling** lemmas, since we couple the real and FL gradient, pseudo and real gradient respectively.

Lemma 5.4 (Bound the difference between real gradient and FL gradient, informal version of Lemma E.4)**.** *With probability $\geq 1-\exp(-\Omega(m^{c_0}))$, for iterations t satisfying $\|U(t) - U(0)\|_{2,\infty} \leq 1/o(m)$, the gradients satisfy*

$$\|\nabla(f,t) - \widetilde{\nabla}(f,t)\|_{2,1} \leq o(m).$$

Lemma 5.5 (Bound the difference between pseudo gradient and real gradient, informal version of Lemma E.5)**.** *With probability $\geq 1-\exp(-\Omega(m^{c_0}))$, for iterations t satisfying $\|U(t) - U(0)\|_{2,\infty} \leq 1/o(m)$, the gradients satisfy*

$$\|\nabla(g,t) - \nabla(f,t)\|_{2,1} \lesssim NJ \cdot o(m).$$

The above two gradient coupling lemmas are essential in proving Theorem 5.6, which is our convergence result.

Theorem 5.6 (Convergence result, informal version of Theorem E.3)**.** *Let $R \geq 1$, $\epsilon \in (0,1)$. Let $K \geq 1$ and let $T \geq \mathrm{poly}(R/\epsilon)$. There exists $M = \mathrm{poly}(n, R, 1/\epsilon)$, such that for every $m \geq M$, with probability at least $1 - \exp(-\Omega(m^{c_0}))$, for every U^* satisfying*

$$\|U^* - U(0)\|_{2,\infty} \leq R/m^{c_1},$$

running Algorithm 1 with setting

$$\eta_{\mathrm{global}} = 1/\,\mathrm{poly}(NJ, R, 1/\epsilon) \quad \textit{and} \quad \eta_{\mathrm{local}} = 1/K$$

will output weights $\{U(t)\}_{t=1}^T$ that satisfy

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{\mathcal{A}}\left(f_{U(t)}\right) \leq \mathcal{L}_{\mathcal{A}^*}\left(f_{U^*}\right) + \epsilon.$$

In the proof of Theorem 5.6 we first bound the local gradient $\nabla_r(f_c, t, k)$. We first consider the pseudo-network, split the difference into three part and bound the three quantities respectively. Namely, we have

$$\mathcal{L}(g_{U(t)}, S(t)) - \mathcal{L}(g_{U^*}, S(t)) \leq \alpha(t) + \beta(t) + \gamma(t),$$

where we denote

$$\begin{cases} \alpha(t) := \langle \widetilde{\nabla}(f,t), U(t) - U^* \rangle; \\ \beta(t) := \|\nabla(f,t) - \widetilde{\nabla}(f,t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty}; \\ \gamma(t) := \|\nabla(g,t) - \nabla(f,t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty}. \end{cases}$$

In bounding $\alpha(t)$, we unfold $\|U(t+1) - U^*\|_F^2$ and have

$$\begin{aligned} \alpha(t) &= \frac{\eta_{\mathrm{global}}}{2}\|\Delta U(t)\|_F^2 \\ &+ \frac{1}{2\eta_{\mathrm{global}}} \cdot (\|U(t) - U^*\|_F^2 - \|U(t+1) - U^*\|_F^2). \end{aligned}$$

We bound $\|\Delta U(t)\|_F^2$ by $\|\Delta U(t)\|_F^2 \leq \eta_{\mathrm{local}} K \cdot o(m)$, and by doing summation over t, we have

$$\begin{aligned} &\sum_{t=1}^{T}\alpha(t) \\ &= \frac{\eta_{\mathrm{global}}}{2}\sum_{t=1}^{T}\|\Delta U(t)\|_F^2 \\ &+ \frac{1}{2\eta_{\mathrm{global}}} \cdot (\|U(1) - U^*\|_F^2 - \|U(T+1) - U^*\|_F^2) \\ &\leq \frac{\eta_{\mathrm{global}}}{2}\sum_{t=1}^{T}\|\Delta U(t)\|_F^2 + \frac{1}{2\eta_{\mathrm{global}}} \cdot \|U(1) - U^*\|_F^2 \\ &\lesssim \eta_{\mathrm{global}}\eta_{\mathrm{local}} TK \cdot o(m) + \frac{1}{\eta_{\mathrm{global}}} m D_{U^*}^2. \end{aligned} \tag{3}$$

In bounding $\beta(t)$, we apply Lemma 5.4 and have

$$\begin{aligned} \beta(t) &= \|\nabla(f,t) - \widetilde{\nabla}(f,t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty} \\ &\lesssim o(m) \cdot \|U(t) - U^*\|_{2,\infty} \\ &\lesssim o(m) \cdot (\|U(t) - U(0)\|_{2,\infty} + D_{U^*}) \end{aligned}$$

where $D_{U^*} := \|U^* - U(0)\|_{2,\infty} \leq R/m^{c_1}$. As for the first term, we use the following bound

$$\begin{aligned}
&\|U(t) - U(0)\|_{2,\infty} \\
&\leq \eta_{\text{global}} \sum_{\tau=1}^{t} \|\Delta U(\tau)\|_{2,\infty} \\
&= \eta_{\text{global}} \sum_{\tau=1}^{t} \|\frac{\eta_{\text{local}}}{N} \sum_{c=1}^{N} \sum_{k=0}^{K-1} \nabla(f_c, t, k)\|_{2,\infty} \\
&\leq \frac{\eta_{\text{global}}\eta_{\text{local}}}{N} \sum_{\tau=1}^{t} \sum_{c=1}^{N} \sum_{k=0}^{K-1} \|\nabla(f_c, t, k)\|_{2,\infty} \\
&\leq \eta_{\text{global}}\eta_{\text{local}} tKm^{-1/3}
\end{aligned}$$

and have $\beta(t) \lesssim \eta_{\text{global}}\eta_{\text{local}} tK \cdot o(m) + o(m) \cdot D_{U^*}$. Then we do summation over t and obtain

$$\sum_{t=1}^{T} \beta(t) \lesssim \eta_{\text{global}}\eta_{\text{local}} T^2 K \cdot o(m) + o(m) \cdot T D_{U^*}. \tag{4}$$

In bounding $\gamma(t)$, we apply Lemma 5.5 and have

$$\begin{aligned}
\gamma(t) = &\ \|\nabla(g, t) - \nabla(f, t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty} \\
\lesssim &\ NJ \cdot o(m) \cdot (\|U(t) - U(0)\|_{2,\infty} + D_{U^*}).
\end{aligned}$$

Then we do summation over t and have

$$\begin{aligned}
\sum_{t=1}^{T} \gamma(t) \lesssim &\ \eta_{\text{global}}\eta_{\text{local}} T^2 KNJ \cdot o(m) \\
&+ TNJ \cdot o(m) D_{U^*}.
\end{aligned} \tag{5}$$

Putting Eq.(3), Eq.(4) and Eq.(5) together with our choice of all parameters (i.e. $\eta_{\text{local}}, \eta_{\text{global}}, R, K, T, m$), we obtain

$$\begin{aligned}
&\frac{1}{T} \sum_{\tau=1}^{T} \mathcal{L}(g_{U(\tau)}, S(\tau)) - \frac{1}{T} \sum_{\tau=1}^{T} \mathcal{L}(g_{U^*}, S(\tau)) \\
&\leq \frac{1}{T} (\sum_{\tau=1}^{T} \alpha(\tau) + \sum_{\tau=1}^{T} \beta(\tau) + \sum_{t=1}^{T} \gamma(\tau)) \leq O(\epsilon).
\end{aligned}$$

From Theorem D.2 in appendix, we have $\sup_{x \in \mathcal{X}} |f_U(x) - g_U(x)| \leq O(\epsilon)$, and thus,

$$\frac{1}{T} \sum_{t=1}^{T} \mathcal{L}(f_{U(t)}, \mathcal{S}(t)) - \mathcal{L}(f_{U^*}, \mathcal{S}(t)) \leq O(\epsilon). \tag{6}$$

From the definition of $\mathcal{A}^*$ we have $\mathcal{L}(f_{U^*}, S(t)) \leq \mathcal{L}_{\mathcal{A}^*}(f_{U^*})$. From the definition of loss function we have $\mathcal{L}(f_{U(t)}, \mathcal{S}(t)) = \mathcal{L}_{\mathcal{A}}(f_{U(t)})$.

Moreover, since Eq. (6) holds for all $\epsilon > 0$, we can replace $O(\epsilon)$ with ϵ. Thus we prove for all $\epsilon > 0$,

$$\frac{1}{T} \sum_{t=1}^{T} \mathcal{L}_{\mathcal{A}}(f_{U(t)}) \leq \mathcal{L}_{\mathcal{A}^*}(f_{U^*}) + \epsilon.$$

Combining the Results. From Theorem 5.2 we obtain U^* that is close to $U(0)$ (i.e., $\|U^* - U(0)\|_{2,\infty} \leq R/m^{c_1}$) and makes $\mathcal{L}_{\mathcal{A}^*}(f_{U^*})$ close to zero (i.e., $\mathcal{L}_{\mathcal{A}^*}(f_{U^*}) \leq \epsilon$). By using Theorem 5.6 we know the average of $\mathcal{L}_{\mathcal{A}}(f_{U(t)})$ is dominated by $\mathcal{L}_{\mathcal{A}^*}(f_{U^*})$ (i.e., $\frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\mathcal{A}}(f_{U(t)}) \leq \mathcal{L}_{\mathcal{A}^*}(f_{U^*}) + \epsilon$). By aggregating these two results, we prove that the minimal of $\mathcal{L}_{\mathcal{A}}(f_{U(t)})$ is ϵ small:

$$\begin{aligned}
\min_{t \in [T]} \mathcal{L}_{\mathcal{A}}(f_{U(t)}) &\leq \frac{1}{T} \sum_{t=1}^{T} \mathcal{L}_{\mathcal{A}}\left(f_{U(t)}\right) \\
&\leq \mathcal{L}_{\mathcal{A}^*}(f_{U^*}) + \epsilon \leq 2\epsilon.
\end{aligned}$$

Thus we finish the proof of our main Theorem 4.1. For the full proof and the choice of parameters, we refer readers to the appendix for details.

6. Numerical Results

In this section, we examine our theoretical result (Theorem 4.1) on data separability γ (Definition 3.4), a standard assumption in over-parameterized neural network convergence analysis. We simulate synthetic data with different levels of data separability as shown in Fig. 1. Specifically, each data point contains two dimensions. Each class of data is generated from two Gaussian distributions (std=1) with different means. Each class consists of two Gaussian clusters where intra-class cluster centroids are closer than inter-class distances. We perform binary classification tasks on the simulated datasets using multi-layer perceptrons MLP with one hidden layer with 128 neurons. To increase learning difficulty, 5% of labels are randomly flipped. For each class, we simulated 400 data points as training sets and 100 data points as a testing set. The training data is even divided into four parts to simulate four clients. To simulate different levels of separability, we expand/shrink data features by (2.5, 1.5, 0.85) to construct (large, medium, small) data separability. Note that the whole dataset is not normalized before feeding into the classifier.

We deploy PGD (Madry et al., 2018) to generate adversarial examples during FAL training with the box of radius $\rho = 0.0314$, each perturbation step of 7, and step length of 0.00784. Model aggregation follows FedAvg (McMahan et al., 2017) after each local update. We use SGD optimizer with batch size 50. We depict the training and testing accuracy curves in Fig. 2(a), where solid lines strand for training and dash line stand for testing. The total communication round for is 100, and we observe training convergence for high (blue) and medium (green) separability datasets with learning rate 1e-5. However, a low separability dataset requires a smaller learning rate (i.e., 5e-6) to avoid divergence. From Theorem 4.1 we know a larger data separability γ results in a smaller R, and we can choose a larger learning rate to achieve convergence. Hence, the se-

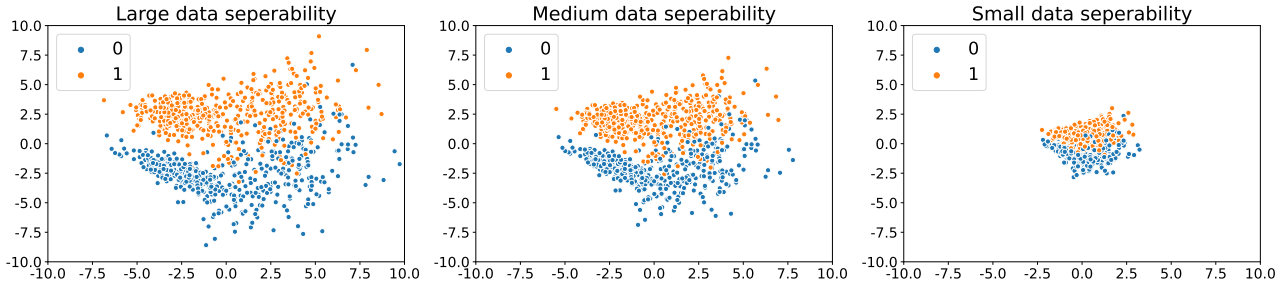
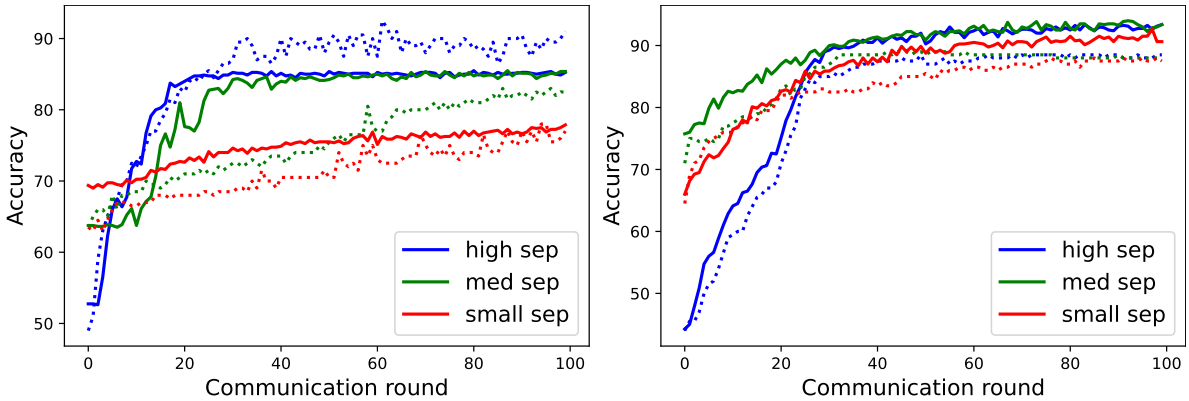


Figure 1: Simulated data with different levels of data separability in numerical experiment.



(a) FAL (b) FedAvg

Figure 2: Training and testing curve on datasets with different levels of data separability. Solid lines present training curves and dash lines present testing curves.

lection of learning rate for small separability is consistent with the constraint of learning rate η_{global} implied in Theorem 4.1. We empirically observe that a dataset with larger data separability γ converges faster with the flexibility of choosing a large learning rate, which is affirmative of our theoretical results that convergence round $T \geq \text{poly}(R/\epsilon)$ has a larger lower bound with a smaller γ, where

$$R = \text{poly}((NJ/\epsilon)^{1/\gamma}).$$

In addition, we compare with the accuracy curves obtained by using FedAvg (McMahan et al., 2017). As shown Fig. 2(b), all the datasets converge at around round 40. Therefore we notice that same data separability scales have larger affect in FAL training.

7. Conclusion

We have studied the convergence of a general format of adopting adversarial training in FL setting to improve FL training robustness. We propose the general framework, FAL, which deploys adversarial samples generation-based adversarial training method on the client-side and then aggregate local model using FedAvg. In FAL, each client is trained via min-max optimization with inner loop adversarial generation and outer loop loss minimization. As far as we know, we are the first to detail the proof of theoretical convergence guarantee for over-parameterized ReLU network on the presented FAL strategy, using gradient descent. Unlike the convergence of adversarial training in classical settings, we consider the updates on both local client and global server sides. Our result indicates that we can control learning rates η_{local} and η_{global} according to the local update steps K and global communication round T to make the minimal loss close to zero. The technical challenges lie in the multiple local update steps and heterogeneous data, leading to the difficulties of convergence. We use gradient coupling methods twice, and together we show the model updates of each global updating bounded in our federated adversarial learning.

Note that we do not require IID assumptions for data distribution. In sum, the proposed FAL formulation and analysis framework can well handle the multi-local updates and non-IID data in FL. Our framework can be generalized to other FL aggregation methods such as sketching and selective aggregation. We denote these generalization in FL aggregation as intriguing future works.

References

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. In *NeurIPS*. https://arxiv.org/pdf/1811.04918.pdf, 2019a.

Allen-Zhu, Z., Li, Y., and Song, Z. On the convergence rate of training recurrent neural networks. In *NeurIPS*. https://arxiv.org/pdf/1810.12065.pdf, 2019b.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *ICML*. https://arxiv.org/pdf/1811.03962.pdf, 2019c.

Alman, J., Liang, J., Song, Z., Zhang, R., and Zhuo, D. Bypass exponential time preprocessing: Fast neural network training via weight-data correlation preprocessing. *arXiv preprint arXiv:2211.14227*, 2022.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. In *NeurIPS*. https://arxiv.org/pdf/1904.11955.pdf, 2019a.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *ICML*. https://arxiv.org/pdf/1901.08584.pdf, 2019b.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

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.

Bhagoji, A. N., Chakraborty, S., Mittal, P., and Calo, S. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pp. 634–643. PMLR, 2019.

Bian, S., Song, Z., and Yin, J. Federated Empirical Risk Minimization via Second-Order Method. *arxiv preprint*, 2023.

Brand, J. v. d., Peng, B., Song, Z., and Weinstein, O. Training (overparametrized) neural networks in near-linear time. In *ITCS*. https://arxiv.org/pdf/2006.11648.pdf, 2021.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57. IEEE, 2017.

Chernoff, H. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, pp. 493–507, 1952.

Cohen, M. B., Lee, Y. T., and Song, Z. Solving linear programs in the current matrix multiplication time. In *STOC*, 2019.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pp. 2206–2216. PMLR, 2020.

Deng, Y., Kamani, M. M., and Mahdavi, M. Distributionally robust federated averaging. *Advances in Neural Information Processing Systems*, 33:15111–15122, 2020.

Deng, Y., Li, Z., and Song, Z. Attention scheme inspired softmax regression. *arXiv preprint arXiv:2304.10411*, 2023.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *ICML*. https://arxiv.org/pdf/1811.03804, 2019a.

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

Gao, R., Cai, T., Li, H., Hsieh, C.-J., Wang, L., and Lee, J. D. Convergence of adversarial training in overparametrized neural networks. *Advances in Neural Information Processing Systems*, 32:13029–13040, 2019.

Gao, Y., Qin, L., Song, Z., and Wang, Y. A sublinear adversarial training algorithm. *arXiv preprint arXiv:2208.05395*, 2022.

Gao, Y., Mahadevan, S., and Song, Z. An overparameterized exponential regression. *arXiv preprint arXiv:2303.16504*, 2023a.

Gao, Y., Song, Z., and Yin, J. An iterative algorithm for rescaled hyperbolic functions regression. *arXiv preprint arXiv:2305.00660*, 2023b.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Gu, Y. and Song, Z. A faster small treewidth sdp solver. *arXiv preprint arXiv:2211.06033*, 2022.

Hazan, E. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Hu, H., Song, Z., Weinstein, O., and Zhuo, D. Training overparametrized neural networks in sublinear time. *arXiv preprint arXiv:2208.04508*, 2022.

Huang, B., Li, X., Song, Z., and Yang, X. Fl-ntk: A neural tangent kernel-based framework for federated learning analysis. In *International Conference on Machine Learning*, pp. 4423–4434. PMLR, 2021.

Huang, B., Jiang, S., Song, Z., Tao, R., and Zhang, R. Solving sdp faster: A robust ipm framework and efficient implementation. In *FOCS*, 2022.

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.

Jiang, H., Kathuria, T., Lee, Y. T., Padmanabhan, S., and Song, Z. A faster interior point method for semidefinite programming. In *FOCS*, 2020.

Jiang, S., Song, Z., Weinstein, O., and Zhang, H. Faster dynamic matrix inverse for faster lps. In *STOC*, 2021.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Lee, J. D., Shen, R., Song, Z., Wang, M., and Yu, Z. Generalized leverage score sampling for neural networks. In *NeurIPS*, 2020.

Lee, Y. T., Song, Z., and Zhang, Q. Solving empirical risk minimization in the current matrix multiplication time. In *COLT*, pp. 2140–2157, 2019.

Li, S., Song, Z., Xia, Y., Yu, T., and Zhou, T. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint*, 2023a.

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, 2020a*, 2020a.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 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., 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*, 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 *NeurIPS*. https://arxiv.org/pdf/1808.01204.pdf, 2018.

Li, Z., Song, Z., and Zhou, T. Solving regularized exp, cosh and sinh regression problems. *arXiv preprint, 2303.15725*, 2023b.

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.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*. https://arxiv.org/pdf/1706.06083.pdf, 2018.

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

Munteanu, A., Omlor, S., Song, Z., and Woodruff, D. Bounding the width of neural networks via coupled initialization a worst case analysis. In *International Conference on Machine Learning*, pp. 16083–16122. PMLR, 2022.

Qin, L., Song, Z., Zhang, L., and Zhuo, D. An online and unified algorithm for projection matrix vector multiplication with application to empirical risk minimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 101–156. PMLR, 2023.

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

Reisizadeh, A., Farnia, F., Pedarsani, R., and Jadbabaie, A. Robust federated learning: The case of affine distribution shifts. *arXiv preprint arXiv:2006.08907*, 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.

Samangouei, P., Kabkab, M., and Chellappa, R. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Song, Z. and Yang, X. Quadratic suffices for over-parametrization via matrix chernoff bound. In *arXiv preprint*. https://arxiv.org/pdf/1906.03593.pdf, 2019.

Song, Z. and Yu, Z. Oblivious sketching-based central path method for linear programming. In *International Conference on Machine Learning*, pp. 9835–9847. PMLR, 2021.

Song, Z., Yang, S., and Zhang, R. Does preprocessing help training over-parameterized neural networks? *Advances in Neural Information Processing Systems*, 34, 2021a.

Song, Z., Zhang, L., and Zhang, R. Training multi-layer over-parametrized neural network in subquadratic time. *arXiv preprint arXiv:2112.07628*, 2021b.

Song, Z., Wang, Y., Yu, Z., and Zhang, L. Sketching for first order method: Efficient algorithm for low-bandwidth channel and vulnerability. In *ICML*, 2023.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *arXiv preprint*. https://arxiv.org/pdf/1312.6199.pdf, 2013.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

Tu, Z., Zhang, J., and Tao, D. Theoretical analysis of adversarial learning: A minimax approach. *arXiv preprint arXiv:1811.05232*, 2018.

Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H. V. Tackling the objective inconsistency problem in heterogeneous federated optimization. *arXiv preprint arXiv:2007.07481*, 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.

Yin, D., Chen, Y., Kannan, R., and Bartlett, P. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pp. 5650–5659. PMLR, 2018.

Zhang, G., Lu, S., Liu, S., Chen, X., Chen, P.-Y., Martie, L., and Horesh, Lior abd Hong, M. Distributed adversarial training to robustify deep neural networks at scale. 2021. URL https://openreview.net/pdf?id=kmBFHJ5pr0o.

Zhang, L. Speeding up optimizations via data structures: Faster search, sample and maintenance. Master's thesis, Carnegie Mellon University, 2022.

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

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 *NeurIPS*. arXiv preprint arXiv:2002.06668, 2020b.

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.

Zizzo, G., Rawat, A., Sinn, M., and Buesser, B. Fat: Federated adversarial training. *arXiv preprint arXiv:2012.01791*, 2020.

Roadmap. The appendix is organized as follows. We introduce the probability tools to be used in our proof in Section A. In addition, we introduce the preliminaries in Section B. We present the proof overview in Section C and additional remarks used in the proof sketch in Section D. We show the detailed proof for the convergence in Section E and the detailed proof of existence in Section F correspondingly.

A. Probability Tools

We introduce the probability tools that will be used in our proof. First we present two lemmas about random variable's tail bound in Lemma A.1 and A.2:

Lemma A.1 (Chernoff bound (Chernoff, 1952))**.** *Let $x = \sum_{i=1}^n x_i$, where $x_i = 1$ with probability p_i and $x_i = 0$ with probability $1 - p_i$, and all x_i are independent. Let $\mu = \mathbb{E}[x] = \sum_{i=1}^n p_i$. Then*
1. $\Pr[x \geq (1+\delta)\mu] \leq \exp(-\delta^2\mu/3)$, $\forall \delta > 0$ *;*
2. $\Pr[x \leq (1-\delta)\mu] \leq \exp(-\delta^2\mu/2)$, $\forall 0 < \delta < 1$.

Lemma A.2 (Bernstein inequality (Bernstein, 1924))**.** *Let $Y_1, \cdots, Y_n$ be independent zero-mean random variables. Suppose that for $i \in [n], |Y_i| \leq M$ almost surely. Then for all $t > 0$, we have*

$$\Pr\left[\sum_{i=1}^n Y_i > t\right] \leq \exp\left(-\frac{t^2/2}{\sum_{i=1}^n \mathbb{E}[Y_i^2] + Mt/3}\right).$$

Next, we introduce Lemma A.3 about CDF of Gaussian distributions:

Lemma A.3. *Let $Z \sim \mathcal{N}(0, \sigma^2)$ denotes a Gaussian random variable, then we have*

$$\Pr[|Z| \leq t] \in \left(\frac{2}{3}\frac{t}{\sigma}, \frac{4}{5}\frac{t}{\sigma}\right).$$

Finally, we introduce Claim A.4 about elementary anti-concentration property of Gaussian distribution.

Claim A.4. *Let $z \sim \mathcal{N}(0, I_d)$ and $u \sim \mathcal{N}(0, 1)$ are independent Gaussian random variables. Then for all $t \geq 0$ and $x \in \mathbb{R}^d$ that satisfies $\|x\|_2 = 1$, we have*

$$\Pr[|\langle x, z\rangle + v| \leq t] = O(t).$$

B. Preliminaries

B.1. Notations

For a vector x, we use $\|x\|_p$ to denote its ℓ_p norm, in this paper we mainly consider the situation when $p = 1, 2$, or ∞.

For a matrix $U \in \mathbb{R}^{d\times m}$, we use $U^\top$ to denote its transpose and use $\mathrm{tr}[U]$ to denote its trace. We use $\|U\|_1$ to denote its entry-wise ℓ_1 norm. We use $\|U\|_2$ to denote its spectral norm. We use $\|U\|_F$ to denote its Frobenius norm. For $j \in [m]$, we let $U_j \in \mathbb{R}^d$ be the j-th column of U. We let $\|U\|_{2,1}$ denotes $\sum_{j=1}^m \|U_j\|_2$. We let $\|U\|_{2,\infty}$ denotes $\max_{j\in[m]} \|U_j\|_2$. For two matrices X and Y, we denote their Euclidean inner product as $\langle X, Y\rangle := \mathrm{tr}[X^\top Y]$.

We denote Gaussian distribution with mean μ and covariance Σ as $\mathcal{N}(\mu, \Sigma)$. We use $\sigma(\cdot)$ to denote the ReLU function, and use $\mathbb{1}\{A\}$ to denote the indicator function of A.

B.2. Two-Layer Neural Network and Initialization

In this paper, we focus on a two-layer neural network that has m neurons in the hidden layer, where each neuron is a ReLU activation function. We define the global network as

$$f_U(x) := \sum_{r=1}^m a_r \cdot \sigma(\langle U_r, x\rangle + b_r) \tag{7}$$

and for $c \in [N]$, we define the local network of client c as

$$f_{W_c}(x) := \sum_{r=1}^m a_r \cdot \sigma(\langle W_{c,r}, x\rangle + b_r). \tag{8}$$

Here $U = (U_1, U_2, \ldots, U_m) \in \mathbb{R}^{d\times m}$ is the global hidden weight matrix, $W_c = (W_{c,1}, \ldots, W_{c,m}) \in \mathbb{R}^{d\times m}$ is the local hidden weight matrix of client c, and $a = (a_1, a_2, \ldots, a_m) \in \mathbb{R}^m$ denotes the output weight, $b = (b_1, b_2, \ldots, b_m) \in \mathbb{R}^m$ denotes the bias. During the process of federated adversarial learning, for convenience we keep a and b equal to their initialized values and only update U and W, so we can write the global network as $f_U(x)$ and the local network as $f_{W_c}(x)$. For the situation we don't care about the weight matrix, we write $f(x)$ or $f_c(x)$ for short. Next, we make some standard assumptions regarding our training set.

Definition B.1 (Dataset)**.** *There are N clients and $n = NJ$ data in total.*[4] *Let $\mathcal{S} = \cup_{c\in[N]}\mathcal{S}_c$ where $\mathcal{S}_c = \{(x_{c,1}, y_{c,1}), ..., (x_{c,J}, y_{c,J})\} \subseteq \mathbb{R}^d \times \mathbb{R}$ denotes the J training data of client c. Without loss of generality, we assume that $\|x_{c,j}\|_2 = 1$ holds for all $c \in [N], j \in [J]$, and the last coordinate of each point equals to $1/2$, so we consider $\mathcal{X} := \{x \in \mathbb{R}^d : \|x\|_2 = 1,\ x_d = 1/2\}$. For simplicity, we assume that $|y_{c,j}| \le 1$ holds for all $c \in [N]$ and $j \in [J]$.*[5]

We now define the initialization for the neural networks.

Definition B.2 (Initialization)**.** *The initialization of $a \in \mathbb{R}^m, U \in \mathbb{R}^{d\times m}, b \in \mathbb{R}^m$ is $a(0) \in \mathbb{R}^m, U(0) \in \mathbb{R}^{d\times m}, b(0) \in \mathbb{R}^m$. The initialization of client c's local weight matrix W_c is $W_c(0, 0) = U(0)$. Here the second term in W_c denotes iteration of local steps.*

- *For each $r \in [m]$, $a_r(0)$ are i.i.d. sampled from $[-1/m^{1/3}, +1/m^{1/3}]$ uniformly.*
- *For each $i \in [d], r \in [m]$, $U_{i,r}(0)$ and $b_r(0)$ are i.i.d. random Gaussians sampled from $\mathcal{N}(0, 1/m)$. Here $U_{i,r}$ means the (i, r)-entry of U.*

For each global iteration $t \in [T]$,

- *For each $c \in [N]$, the initial value of client c's local weight matrix W_c is $W_c(t, 0) = U(t)$.*

[4] For simplicity, we assume that all clients have same number of training data. Our result can be generalized to the setting where each client has a different number of data as the future work.

[5] Our assumptions on data points are reasonable since we can do scale-up. In addition, l_2 norm normalization is a typical technique in experiments. Same assumptions also appears in many previous theoretical works like (Arora et al., 2019b; Allen-Zhu et al., 2019a;b).

B.3. Adversary and Well-Separated Training Sets

We first formulate the adversary as a mapping.

Definition B.3 (ρ-Bounded adversary)**.** *Let $\mathcal{F}$ denote the function class. An adversary is a mapping $\mathcal{A} : \mathcal{F} \times \mathcal{X} \times \mathbb{R} \to \mathcal{X}$ which denotes the adversarial perturbation. For $\rho > 0$, we define the ℓ_2 ball as $\mathcal{B}_2(x, \rho) := \{\widetilde{x} \in \mathbb{R}^d : \|\widetilde{x} - x\|_2 \leq \rho\} \cap \mathcal{X}$, we say an adversary $\mathcal{A}$ is ρ-**bounded** if it satisfies*

$$\mathcal{A}(f, x, y) \in \mathcal{B}_2(x, \rho).$$

*Moreover, given $\rho > 0$, we denote the **worst-case** adversary as $\mathcal{A}^* := \operatorname{argmax}_{\widetilde{x} \in \mathcal{B}_2(x,\rho)} \ell(f(\widetilde{x}), y)$, where ℓ is defined in Definition B.5.*

In the over-parameterized regime, it is a standard assumption that the training set is well-separated. Since we deal with adversarial perturbations, we require the following γ-separability, which is a bit stronger.

Definition B.4 (γ-separability)**.** *Let $\gamma \in (0, 1/2), \delta \in (0, 1/2), \rho \in (0, 1/2)$ denote three parameters such that $\gamma \leq \delta \cdot (\delta - 2\rho)$. We say our training set $\mathcal{S} = \cup_{c \in [N]} \mathcal{S}_c = \cup_{c \in [N], j \in [J]} \{(x_{c,j}, y_{c,j})\} \subset \mathbb{R}^d \times \mathbb{R}$ is **globally γ-separable** w.r.t a ρ-bounded adversary, if*

$$\min_{c_1 \neq c_2, j_1 \neq j_2} \|x_{c_1,j_1} - x_{c_2,j_2}\|_2 \geq \delta.$$

It is noteworthy that our problem setup does not need the assumption on independent and identically distribution (IID) on data, thus such a formation can be applied to unique challenge of the non-IID setting in FL.

B.4. Robust Loss Function

We define the following Lipschitz convex loss function that will be used.

Definition B.5 (Lipschitz convex loss)**.** *A loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is said to be a **Lipschitz convex loss**, if it satisfies the following four properties:*

- *non-negative;*
- *convex in the first input of ℓ;*
- $1-$*Lipshcitz, which means $\|\ell(x_1, y_1) - \ell(x_2, y_2)\|_2 \leq \|(x_1, y_1) - (x_2, y_2)\|_2$;*
- $\ell(y, y) = 0$ *for all $y \in \mathbb{R}$.*

In this paper we assume ℓ is a Lipschitz convex loss. Next, we define our robust loss function of a neural network, which is based on the adversarial examples generated by a ρ-bounded adversary $\mathcal{A}$.

Definition B.6 (Training loss)**.** *Given a client's training set $\mathcal{S}_c = \{(x_{c,j}, y_{c,j})\}_{j=1}^J \subset \mathbb{R}^d \times \mathbb{R}$ of J samples. Let $f_c : \mathbb{R}^d \to \mathbb{R}$ be a net. We define loss to be $\mathcal{L}(f_c, \mathcal{S}_c) := \frac{1}{J} \sum_{j=1}^J \ell\left(f_c(x_{c,j}), y_{c,j}\right)$. Given $\mathcal{S} = \cup_{c \in [N]} \mathcal{S}_c$, the global loss is defined as*

$$\mathcal{L}(f_U, \mathcal{S}) := \frac{1}{NJ} \sum_{c=1}^N \sum_{j=1}^J \ell(f_U(x_{c,j}), y_{c,j}).$$

Given an adversary $\mathcal{A}$ that is ρ-bounded, we define the global loss with respect to $\mathcal{A}$ as

$$\begin{aligned}\mathcal{L}_{\mathcal{A}}(f_U) &:= \frac{1}{NJ} \sum_{c=1}^N \sum_{j=1}^J \ell(f_U(\mathcal{A}(f_c, x_{c,j}, y_{c,j})), y_{c,j}) \\ &= \frac{1}{NJ} \sum_{c=1}^N \sum_{j=1}^J \ell(f_U(\widetilde{x}_{c,j}), y_{c,j})\end{aligned}$$

*and also define the global robust loss (in terms of **worst-case**) as*

$$\begin{aligned}\mathcal{L}_{\mathcal{A}^*}(f_U) &:= \frac{1}{NJ}\sum_{c=1}^{N}\sum_{j=1}^{J}\ell(f_U(\mathcal{A}^*(f_c, x_{c,j}, y_{c,j})), y_{c,j}) \\ &= \frac{1}{NJ}\sum_{c=1}^{N}\sum_{j=1}^{J}\max_{x^*_{c,j}\in\mathcal{B}_2(x_{c,j},\rho)}\ell\left(f_U(x^*_{c,j}), y_{c,j}\right).\end{aligned}$$

Moreover, since we deal with pseudo-net which is defined in Definition D.1, we also define the loss of a pseudo-net as $\mathcal{L}(g_c, \mathcal{S}_c) := \frac{1}{J}\sum_{j=1}^{J}\ell\left(g_c(x_{c,j}), y_{c,j}\right)$ *and* $\mathcal{L}(g_U, \mathcal{S}) := \frac{1}{NJ}\sum_{c=1}^{N}\sum_{j=1}^{J}\ell(g_U(x_{c,j}), y_{c,j})$.

B.5. Federated Adversarial Learning Algorithm

Classical adversarial training algorithm can be found in (Zhang et al., 2020b). Different from the classical setting, our federated adversarial learning of a local neural network f_{W_c} against an adversary $\mathcal{A}$ is shown in Algorithm 2, where there are two procedures: one is CLIENTUPDATE running on client side and the other is SERVEREXECUTION running on server side. These two procedures are iteratively processed through communication iterations. Adversarial training is addressed in procedure CLIENTUPDATE. Hence, there are two loops in CLIENTUPDATE procedure: the outer loop is iteration for local model updating; and the inner loop is iteratively generating adversarial samples by the adversary $\mathcal{A}$. In the outer loop in SERVEREXECUTION procedure, the neural network's parameters are updated to reduce its prediction loss on the new adversarial samples. These loops constitute an intertwining dynamics.

Algorithm 2 Federated Adversarial Learning (FAL). Complete and formal version of Algorithm 1.

/*Defining notations and parameters*/
We use c to denote the client's index
The training set of client c is denoted as $\mathcal{S}_c = \{(x_{c,j}, y_{c,j})\}_{j=1}^{J}$
Let $\mathcal{A}$ be the adversary
We denote local learning rate as η_{local}
We denote global learning rate as η_{global}
We denote local updating iterations as K
We denote global communication round as T

/*Initialization*/
Initialization $a(0) \in \mathbb{R}^m, U(0) \in \mathbb{R}^{d\times m}, b(0) \in \mathbb{R}^m$
For $t = 0 \to T$, we iteratively run **Procedure A** then **Procedure B**

/* Procedure running on client side */
procedure A. CLIENTUPDATE(t, c)
$\mathcal{S}_c(t) \leftarrow \emptyset$
$W_c(t, 0) \leftarrow U(t)$ ▷ Receive global model weights update
for $k = 0 \to K - 1$ **do**
for $j = 1 \to J$ **do**
$\widetilde{x}_{c,j}^{(t)} \leftarrow \mathcal{A}(f_{W_c(t,k)}, x_{c,j}, y_{c,j})$ ▷ Adversarial examples, f_{W_c} is defined as (8)
$\mathcal{S}_c(t) \leftarrow \mathcal{S}_c(t) \cup (\widetilde{x}_{c,j}^{(t)}, y_{c,j})$
end for
$W_c(t, k+1) \leftarrow W_c(t, k) - \eta_{\text{local}} \cdot \nabla_{W_c}\mathcal{L}(f_{W_c(t,k)}, \mathcal{S}_c(t))$
end for
$\Delta U_c(t) \leftarrow W_c(t, K) - U(t)$
Send $\Delta U_c(t)$ to SERVEREXECUTION
end procedure

/*Procedure running on server side*/
procedure B. SERVEREXECUTION(t):
for each client c **in parallel do**
$\Delta U_c(t) \leftarrow$ CLIENTUPDATE(c, t) ▷ Receive local model weights update
$\Delta U(t) \leftarrow \frac{1}{N}\sum_{c\in[N]} \Delta U_c(t)$
$U(t+1) \leftarrow U(t) + \eta_{\text{global}} \cdot \Delta U(t)$ ▷ Aggregation on the server side
Send $U(t+1)$ to client c for CLIENTUPDATE(c, t)
end for
end procedure

C. Proof Overview

In this section we give an overview of our main result's proof. Two theorems to be used are Theorem E.3 and Theorem F.3.

C.1. Pseudo-Network

We study the over-parameterized neural nets' well-approximated pseudo-network to learn gradient descent for over-parameterized neural nets whose weights are close to initialization. The introducing of pseudo-network makes the proof more intuitive.

To be specific, we give the definition of pseudo-network in Section D, and also state Theorem D.2 which shows the fact that the pseudo-network approximates the real network uniformly well. It can be seen that the notion of pseudo-network is used for several times in our proof.

C.2. Online Gradient Descent in Federated Adversarial Learning

Our federated adversarial learning algorithm is formulated in *online gradient descent* framework: at each local step k on the client side, firstly the adversary generates adversarial samples and computes the loss function $\mathcal{L}\left(f_{W_c(t,k)}, \mathcal{S}_c(t)\right)$, then the local client learner takes the fresh loss function and update $W_c(t,k+1) = W_c(t,k) - \eta_{\text{local}} \cdot \nabla_{W_c}\mathcal{L}\left(f_{W_c(t,k)}, \mathcal{S}_c(t)\right)$. We refer our readers to (Gao et al., 2019; Hazan, 2016) for more details regarding online learning and online gradient descent.

Compared with the centralized setting, the key difficulties in the convergence analysis of FL are induced by multiple local step updates of the client side and the step updates on both local and global sides. Specifically, local updates are not the standard gradient as the centralized adversarial training when $K \geq 2$. We used $-\Delta U(t)$ in substitution of the real gradient of U to update the value of $U(t)$. This brings in challenges to bound the gradient of the neural networks. Nevertheless, gradient bounding is challenging in adversarial training solely. We use gradient coupling method twice to solve this core problem: firstly we bound the difference between real gradient and FL gradient in Lemma E.4, then we bound the difference between pseudo gradient and real gradient in Lemma E.5. We show the connection of online gradient descent and federated adversarial learning in the proof of Theorem E.3.

C.3. Existence of Robust Network Near Initialization

In Section F we show that there exists a global network f_{U^*} whose weight is close to the initial value $U(0)$ and makes the worst-case global loss $\mathcal{L}_{\mathcal{A}^*}(f_{U^*})$ sufficiently small. We show that the required width m is $\text{poly}(d, (NJ/\epsilon)^{1/\gamma})$.

Suppose we are given a ρ-bounded adversary. For a globally γ-separable training set, to prove Theorem F.3, first we state Lemma F.1 which shows the existence of function f^* that has "low complexity" and satisfies $f^*(\widetilde{x}_{c,j}) \approx y_{c,j}$ for all data point $(x_{c,j}, y_{c,j})$ and perturbation inputs $\widetilde{x}_{c,j} \in \mathcal{B}_2(x_{c,j}, \rho)$.

Then, we state Lemma F.2 which shows the existence of a pseudo-network g_{U^*} that approximates f^* well. Finally, by using Theorem D.2 we show that f_{U^*} approximates g_{U^*} well. By combining these results, we we finish the proof of Theorem F.3.

D. Real Approximates Pseudo

To make additional remark to proof sketch in Section 5, in this section, we state a tool that will be used in our proof that is related to our definition of pseudo-network. First, we recall the definition of pseudo-network.

Definition D.1 (Pseudo-network)**.** *Given weights $U \in \mathbb{R}^{d\times m}$, $a \in \mathbb{R}^m$ and $b \in \mathbb{R}^m$, the global neural network function $f_U : \mathbb{R}^d \to \mathbb{R}$ is defined as*

$$f_U(x) := \sum_{r=1}^{m} a_r \cdot \sigma(\langle U_r, x\rangle + b_r).$$

Given this $f_U(x)$, we define the corresponding pseudo-network function $g_U : \mathbb{R}^d \to \mathbb{R}$ as

$$g_U(x) := \sum_{r=1}^{m} a_r \cdot \langle U_r(t) - U_r(0), x\rangle \cdot \mathbb{1}\{\langle U_r(0), x\rangle + b_r \geq 0\}.$$

From the definition we can know that pseudo-network can be seen as a linear approximation of the two layer ReLU network we study near initialization. Next, we cite a Theorem from (Zhang et al., 2020b), which gives a uniform bound of the difference between a network and its pseudo-network.

Theorem D.2 (Uniform approximation, Theorem 5.1 in (Zhang et al., 2020b))**.** *Suppose $R \geq 1$ is a constant. Let $\rho := \exp(-\Omega(m^{1/3}))$. As long as $m \geq \mathrm{poly}(d)$, with prob. $1-\rho$, for every $U \in \mathbb{R}^{d\times m}$ satisfying $\|U - U(0)\|_{2,\infty} \leq R/m^{2/3}$, we have $\sup_{x\in\mathcal{X}} |f_U(x) - g_U(x)|$ is at most $O(R^2/m^{1/6})$.*

The randomness is due to initialization.

E. Convergence

Table 1: List of theorems and lemmas in Section E. The main result of this section is Theorem E.3. By saying "Statements Used" we mean these statements are used in the proof in the corresponding section. For example, Lemma E.4, E.5 and Theorem D.2 are used in the proof of Theorem E.3.

Section	Statement	Comment	Statements Used
E.1	Definition E.1 and E.2	Definition	-
E.2	Theorem E.3	Convergence result	Lem. E.4, E.5, Thm. D.2
E.3	Lemma E.4	Approximates real gradient	-
E.4	Lemma E.5	Approximates pseudo gradient	Claim E.6
E.5	Claim E.6	Auxiliary bounding	Claim A.4

E.1. Definitions and Notations

In Section E, we follow the notations used in Definition D.1. Since we are dealing with pseudo-network, we first introduce some additional definitions and notations regarding gradient. For convenience we denote $\widetilde{U} = U(0)$ as the initialization of global weights U.

Definition E.1 (Gradient)**.** *For a local real network $f_{W_c(t,k)}$, we denote its gradient by*

$$\nabla(f_c, t, k) := \nabla_{W_c}\mathcal{L}(f_{W_c(t,k)}, \mathcal{S}_c(t)).$$

If the corresponding pseudo-network is $g_{W_c(t,k)}$, then we define the pseudo-network gradient as

$$\nabla(g_c, t, k) := \nabla_{W_c}\mathcal{L}(g_{W_c(t,k)}, \mathcal{S}_c(t)).$$

Now we consider the global matrix. For convenience we write $\nabla(f,t) := \nabla_U\mathcal{L}(f_{U(t)}, \mathcal{S}(t))$ and $\nabla(g,t) := \nabla_U\mathcal{L}(g_{U(t)}, \mathcal{S}(t))$. We define the **FL gradient** as $\widetilde{\nabla}(f,t) := -\frac{1}{N}\Delta U(t)$.

Definition E.2 (Distance). *For $U^* \in \mathbb{R}^{d\times m}$ such that $\|U^* - \widetilde{U}\|_{2,\infty} \leq R/m^{3/4}$, we define the following distance for simplicity:*

$$D_{\max} := \max_{t\in[T]} \|\widetilde{U} - U(t)\|_{2,\infty}$$
$$D_{U^*} := \|\widetilde{U} - U^*\|_{2,\infty}$$

We have $D_{U^*} = O(R/m^{3/4})$ and $\|U(t) - U^*\|_{2,\infty} \leq D_{\max} + D_{U^*}$ by using triangle inequality.

Table 2: Notations of global model weights in federated learning to be used in this section.

Notation	Meaning	Satisfy
$U(0)$ or $\widetilde{U}$	Initialization of U	$W_c(0,0) = U(0)$
$U(t)$	The value of U after t iterations	$D_{\max} = \max \|U(t) - \widetilde{U}\|_{2,\infty}$
U^*	The value of U after small perturbations from $\widetilde{U}$	$\|U^* - \widetilde{U}\|_{2,\infty} \leq R/m^{3/4}$

E.2. Convergence Result

We are going to prove Theorem E.3 in this section.

Theorem E.3 (Convergence, formal version of Theorem 5.6)**.** *Let $R \geq 1$. Suppose $\epsilon \in (0,1)$. Let $K \geq 1$. Let $T \geq \mathrm{poly}(R/\epsilon)$. There is $M = \mathrm{poly}(n, R, 1/\epsilon)$, such that for every $m \geq M$, with probability $\geq 1 - \exp(-\Omega(m^{1/3}))$, if we run Algorithm 2 by setting*

$$\eta_{\mathrm{global}} = 1/\mathrm{poly}(NJ, R, 1/\epsilon) \quad \textit{and} \quad \eta_{\mathrm{local}} = 1/K,$$

then for every U^* such that $\|U^* - U(0)\|_{2,\infty} \leq R/m^{3/4}$, the output weights $(U(t))_{t=1}^T$ satisfy

$$\frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\mathcal{A}}\left(f_{U(t)}\right) \leq \mathcal{L}_{\mathcal{A}^*}\left(f_{U^*}\right) + \epsilon.$$

The randomness is from $a(0) \in \mathbb{R}^m$, $U(0) \in \mathbb{R}^{d\times m}$, $b(0) \in \mathbb{R}^m$.

Proof. We set our parameters as follows:

$$M = \Omega\Big(\max\big\{(NJ)^8, (\frac{R}{\epsilon})^{12}\big\}\Big)$$
$$\eta_{\mathrm{global}} = O(\frac{\epsilon}{Nm^{1/3}\cdot \mathrm{poly}(R/\epsilon)})$$
$$\eta_{\mathrm{local}} = 1/K$$

Since the loss function is 1-Lipschitz, we first bound the ℓ_2 norm of real net gradient:

$$\|\nabla_r(f_c, t, k)\|_2 \leq |a_r| \cdot \Big(\frac{1}{J}\sum_{j=1}^{J} \sigma'(\langle W_{c,r}(t,k), x_{c,j}\rangle + b_r) \cdot \|\widetilde{x}_{c,j}\|_2\Big) \leq |a_r| \leq \frac{1}{m^{1/3}}. \tag{9}$$

Now we consider the pseudo-net gradient. The loss $\mathcal{L}(g_U, \mathcal{S}(t))$ is convex in U due to the fact that g is linear with U. Then we have

$$\begin{aligned}
&\mathcal{L}(g_{U(t)}, \mathcal{S}(t)) - \mathcal{L}(g_{U^*}, \mathcal{S}(t)) \\
\leq\ & \langle \nabla_U \mathcal{L}(g_{U(t)}, \mathcal{S}(t)), U(t) - U^*\rangle \\
=\ & \langle \widetilde{\nabla}(f,t), U(t) - U^*\rangle + \langle \nabla(f,t) - \widetilde{\nabla}(f,t), U(t) - U^*\rangle + \langle \nabla(g,t) - \nabla(f,t), U(t) - U^*\rangle \\
\leq\ & \alpha(t) + \beta(t) + \gamma(t)
\end{aligned}$$

where the last step follows from

$$\begin{aligned}
\alpha(t) &:= \langle \widetilde{\nabla}(f,t), U(t) - U^* \rangle, \\
\beta(t) &:= \|\nabla(f,t) - \widetilde{\nabla}(f,t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty}, \\
\gamma(t) &:= \|\nabla(g,t) - \nabla(f,t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty}.
\end{aligned}$$

Note that the FL gradient $\widetilde{\nabla}(f,t) = -\frac{1}{N}\Delta U(t)$ is the direction moved by center, in contrast, $\nabla(f,t)$ is the true gradient of function f. We deal with these three terms separately. As for $\alpha(t)$, we have

$$\begin{aligned}
\|U(t+1) - U^*\|_F^2 &= \|U(t) + \eta_{\text{global}}\Delta U(t) - U^*\|_F^2 \\
&= \|U(t) - U^*\|_F^2 - 2N\eta_{\text{global}}\alpha(t) + \eta_{\text{global}}^2\|\Delta U(t)\|_F^2
\end{aligned}$$

and by rearranging the equation we get

$$\alpha(t) = \frac{\eta_{\text{global}}}{2N}\|\Delta U(t)\|_F^2 + \frac{1}{2N\eta_{\text{global}}} \cdot (\|U(t) - U^*\|_F^2 - \|U(t+1) - U^*\|_F^2).$$

Next, we need to upper bound $\|\Delta U(t)\|_F^2$,

$$\begin{aligned}
\|\Delta U(t)\|_F^2 &= \|\frac{\eta_{\text{local}}}{N}\sum_{c=1}^{N}\sum_{k=0}^{K-1}\nabla(f_c,t,k)\|_F^2 \\
&\leq \frac{\eta_{\text{local}}}{N}\sum_{c=1}^{N}\sum_{k=0}^{K-1}\sum_{r=1}^{m}\|\nabla_r(f_c,t,k)\|_2^2 \\
&= \eta_{\text{local}}Km^{1/3} \\
&= m^{1/3}.
\end{aligned} \tag{10}$$

where the last step follows from $K\eta_{\text{local}} = 1$. Then we do summation over t and have

$$\begin{aligned}
\sum_{t=1}^{T}\alpha(t) &= \frac{\eta_{\text{global}}}{2N}\sum_{t=1}^{T}\|\Delta U(t)\|_F^2 + \frac{1}{2N\eta_{\text{global}}} \cdot \sum_{t=1}^{T}(\|U(t) - U^*\|_F^2 - \|U(t+1) - U^*\|_F^2) \\
&= \frac{\eta_{\text{global}}}{2N}\sum_{t=1}^{T}\|\Delta U(t)\|_F^2 + \frac{1}{2N\eta_{\text{global}}} \cdot (\|U(1) - U^*\|_F^2 - \|U(T+1) - U^*\|_F^2) \\
&\leq \frac{\eta_{\text{global}}}{2N}\sum_{t=1}^{T}\|\Delta U(t)\|_F^2 + \frac{1}{2N\eta_{\text{global}}} \cdot \|U(1) - U^*\|_F^2 \\
&\lesssim \frac{\eta_{\text{global}}}{N}Tm^{1/3} + \frac{1}{N\eta_{\text{global}}}mD_{U^*}^2
\end{aligned}$$

where the last step follows from Eq. (10) and $\|\widetilde{U} - U^*\|_F^2 \leq m \cdot \|\widetilde{U} - U^*\|_{2,\infty}^2 = mD_{U^*}^2$.

As for $\beta(t)$, we apply Lemma E.4 and also triangle inequality and have

$$\begin{aligned}
\beta(t) &= \|\nabla(f,t) - \widetilde{\nabla}(f,t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty} \\
&\lesssim m^{2/3} \cdot \|U(t) - U^*\|_{2,\infty} \\
&\lesssim m^{2/3} \cdot (\|U(t) - \widetilde{U}\|_{2,\infty} + D_{U^*}).
\end{aligned}$$

By using Eq. (9) we bound the size of $\|U(t) - \widetilde{U}\|_{2,\infty}$:

$$\begin{aligned}
\|U(t) - \widetilde{U}\|_{2,\infty} &\leq \eta_{\text{global}} \sum_{\tau=1}^{t} \|\Delta U(\tau)\|_{2,\infty} \\
&= \eta_{\text{global}} \sum_{\tau=1}^{t} \|\frac{\eta_{\text{local}}}{N} \sum_{c=1}^{N} \sum_{k=0}^{K-1} \nabla(f_c, t, k)\|_{2,\infty} \\
&\leq \frac{\eta_{\text{global}}\eta_{\text{local}}}{N} \sum_{\tau=1}^{t} \sum_{c=1}^{N} \sum_{k=0}^{K-1} \|\nabla(f_c, t, k)\|_{2,\infty} \\
&\leq \eta_{\text{global}}\eta_{\text{local}} t K m^{-1/3}
\end{aligned}$$

and have

$$\beta(t) \lesssim \eta_{\text{global}}\eta_{\text{local}} t K m^{1/3} + m^{2/3} D_{U^*}.$$

Then we do summation over t and have

$$\begin{aligned}
\sum_{t=1}^{T} \beta(t) &\lesssim \sum_{t=1}^{T} (\eta_{\text{global}}\eta_{\text{local}} t K m^{1/3} + m^{2/3} D_{U^*}) \\
&\lesssim \eta_{\text{global}}\eta_{\text{local}} T^2 K m^{1/3} + m^{2/3} T D_{U^*} \\
&\lesssim \eta_{\text{global}} T^2 m^{1/3} + m^{2/3} T D_{U^*}.
\end{aligned}$$

As for $\gamma(t)$, we apply Lemma E.5 and have

$$\begin{aligned}
\gamma(t) &= \|\nabla(g, t) - \nabla(f, t)\|_{2,1} \cdot \|U(t) - U^*\|_{2,\infty} \\
&\lesssim N J m^{13/24} \cdot (\|U(t) - \widetilde{U}\|_{2,\infty} + D_{U^*}).
\end{aligned}$$

Since $\|U(t) - \widetilde{U}\|_{2,\infty} \leq \eta_{\text{global}}\eta_{\text{local}} t K m^{-1/3}$, we have

$$\gamma(t) \lesssim \eta_{\text{global}}\eta_{\text{local}} t K N J m^{5/24} + N J m^{13/24} D_{U^*}.$$

Then we do summation over t and have

$$\begin{aligned}
\sum_{t=1}^{T} \gamma(t) &\lesssim \sum_{t=1}^{T} \left(\eta_{\text{global}}\eta_{\text{local}} t K N J m^{5/24} + N J m^{13/24} D_{U^*}\right) \\
&\lesssim \eta_{\text{global}}\eta_{\text{local}} T^2 K N J m^{5/24} + N J m^{13/24} T D_{U^*} \\
&\lesssim \eta_{\text{global}} T^2 N J m^{5/24} + N J m^{13/24} T D_{U^*}.
\end{aligned}$$

Next we put it altogether. Note that $D_{U^*} = O(\frac{R}{m^{3/4}})$, thus we obtain

$$\begin{aligned}
&\sum_{t=1}^{T} \mathcal{L}(g_{U(t)}, \mathcal{S}(t)) - \sum_{t=1}^{T} \mathcal{L}(g_{U^*}, \mathcal{S}(t)) \\
\leq &\sum_{t=1}^{T} \alpha(t) + \sum_{t=1}^{T} \beta(t) + \sum_{t=1}^{T} \gamma(t) \\
\lesssim &\frac{\eta_{\text{global}}}{N} T m^{1/3} + \frac{1}{N\eta_{\text{global}}} m D_{U^*}^2 + \eta_{\text{global}} T^2 m^{1/3} \\
&+ m^{2/3} T D_{U^*} + \eta_{\text{global}} T^2 N J m^{5/24} + N J m^{13/24} T D_{U^*} \\
\lesssim &\frac{\eta_{\text{global}}}{N} T m^{1/3} + \frac{1}{N\eta_{\text{global}}} R^2 m^{-1/2} + \eta_{\text{global}} T^2 m^{1/3} \\
&+ R T m^{-1/12} + \eta_{\text{global}} T^2 N J m^{5/24} + N J m^{-5/24} R T.
\end{aligned}$$

We then have

$$
\begin{aligned}
&\frac{1}{T}\sum_{\tau=1}^{T}\mathcal{L}(g_{U(\tau)},\mathcal{S}(\tau)) - \frac{1}{T}\sum_{\tau=1}^{T}\mathcal{L}(g_{U^*},\mathcal{S}(\tau)) \\
\lesssim\ & \frac{\eta_{\text{global}}}{N}m^{1/3} + \frac{1}{N\eta_{\text{global}}T}R^2m^{-1/2} + \eta_{\text{global}}Tm^{1/3} + Rm^{-1/12} \\
&+ \eta_{\text{global}}TNJm^{5/24} + NJm^{-5/24}R. \\
\lesssim\ & \frac{1}{N\eta_{\text{global}}T}R^2m^{-1/2} + \eta_{\text{global}}Tm^{1/3} + Rm^{-1/12} + \eta_{\text{global}}TNJm^{5/24} + NJm^{-5/24}R \\
\leq\ & O(\epsilon).
\end{aligned}
\tag{11}
$$

From Theorem D.2 we know

$$\sup_{x\in\mathcal{X}}|f_U(x) - g_U(x)| \leq O(R^2/m^{1/6}) = O(\epsilon).$$

In addition,

$$\frac{1}{T}\sum_{t=1}^{T}\big(\mathcal{L}(f_{U(t)},\mathcal{S}(t)) - \mathcal{L}(f_{U^*},\mathcal{S}(t))\big) \leq O(\epsilon) \tag{12}$$

From the definition of $\mathcal{A}^*$ we have $\mathcal{L}(f_{U^*},S(t)) \leq \mathcal{L}_{\mathcal{A}^*}(f_{U^*})$. From the definition of loss we have $\mathcal{L}(f_{U(t)},\mathcal{S}(t)) = \mathcal{L}_{\mathcal{A}}(f_{U(t)})$. Moreover, since Eq. (12) holds for all $\epsilon > 0$, we can replace $\frac{\epsilon}{c}$ with ϵ. Thus we prove that for $\forall\epsilon > 0$,

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{\mathcal{A}}(f_{U(t)}) \leq \mathcal{L}_{\mathcal{A}^*}(f_{U^*}) + \epsilon.$$

□

E.3. Approximates Real Global Gradient

We are going to prove Lemma E.4 in this section.

Lemma E.4 (Bounding the difference between real gradient and FL gradient)**.** *Let $\rho := \exp(-\Omega(m^{1/3}))$ With probability $\geq 1-\rho$, for iterations t satisfying*

$$|U(t) - U(0)\|_{2,\infty} \leq O(m^{-15/24}),$$

the following holds:

$$\|\nabla(f,t) - \widetilde{\nabla}(f,t)\|_{2,1} \leq O(m^{2/3}).$$

The randomness is from $a(\tau)\in\mathbb{R}^m$, $U(\tau)\in\mathbb{R}^{d\times m}$, $b(\tau)\in\mathbb{R}^m$ for τ at 0.

Proof. Notice that $\nabla(f,t) = \nabla_U\mathcal{L}(f_{U(t)},\mathcal{S}(t))$ and

$$\widetilde{\nabla}(f,t) = -\frac{1}{N}\Delta U(t) = -\frac{1}{N}\sum_{c=1}^{N}\Delta U_c(t) = \frac{\eta_{\text{local}}}{N}\sum_{c=1}^{N}\sum_{k=0}^{K-1}\nabla(f_c,t,k).$$

So we have

$$\begin{aligned}
\|\nabla(f,t)-\widetilde{\nabla}(f,t)\|_{2,1} &= \sum_{r=1}^{m}\|\nabla_r(f,t)-\widetilde{\nabla}_r(f,t)\|_2 \\
&= \frac{1}{N}\sum_{r=1}^{m}\|N\cdot\nabla_r(f,t)-\eta_{\text{local}}\sum_{c=1}^{N}\sum_{k=0}^{K-1}\nabla_r(f_c,t,k)\|_2 \\
&\leq \frac{\eta_{\text{local}}}{N}\sum_{r=1}^{m}\sum_{k=0}^{K-1}\|\frac{N\cdot\nabla_r(f,t)}{K\eta_{\text{local}}}-\sum_{c=1}^{N}\nabla_r(f_c,t,k)\|_2 \\
&= \frac{1}{NK}\sum_{r=1}^{m}\sum_{k=0}^{K-1}\|N\cdot\nabla_r(f,t)-\sum_{c=1}^{N}\nabla_r(f_c,t,k)\|_2
\end{aligned}$$

where the last step follows from the assumption that $\eta_{\text{local}}=\frac{1}{K}$.

As for $\|N\cdot\nabla_r(f,t)-\sum_{c=1}^{N}\nabla_r(f_c,t,k)\|_2$, we have

$$\begin{aligned}
&\|N\cdot\nabla_r(f,t)-\sum_{c=1}^{N}\nabla_r(f_c,t,k)\|_2 \\
\leq\ & |a_r|\cdot\Big|\big(\frac{N}{NJ}\sum_{c=1}^{N}\sum_{j=1}^{J}\mathbb{1}\{\langle U_r(t),x_{c,j}\rangle+b_r\geq 0\} \\
&-\frac{1}{J}\sum_{c=1}^{N}\sum_{j=1}^{J}\mathbb{1}\{\langle W_{c,r}(t,k),x_{c,j}\rangle+b_r\geq 0\}\big)\cdot\|x_{c,j}\|_2\Big| \\
\leq\ & \frac{1}{m^{1/3}}\cdot\frac{1}{J}\sum_{c=1}^{N}\sum_{j=1}^{J}|\,\mathbb{1}\{\langle U_r(t),x_{c,j}\rangle+b_r\geq 0\}-\mathbb{1}\{\langle W_{c,r}(t,k),x_{c,j}\rangle+b_r\geq 0\}| \\
\leq\ & \frac{N}{m^{1/3}}.
\end{aligned}$$

Then we do summation and have

$$\begin{aligned}
\|\nabla(f,t)-\widetilde{\nabla}(f,t)\|_{2,1} &\leq \frac{1}{NK}\sum_{r=1}^{m}\sum_{k=0}^{K-1}\|N\cdot\nabla_r(f,t)-\sum_{c=1}^{N}\nabla_r(f_c,t,k)\|_2 \\
&\leq \frac{1}{NK}\sum_{r=1}^{m}\sum_{k=0}^{K-1}\frac{N}{m^{1/3}} \\
&= m^{2/3}.
\end{aligned}$$

Thus we finish the proof. □

E.4. Approximates Pseudo Global Gradient

We are going to prove Lemma E.5 in this section.

Lemma E.5. *Let $\rho:=\exp(-\Omega(m^{1/3}))$. With probability $\geq 1-\rho$, for iterations t satisfying*

$$\|U(t)-U(0)\|_{2,\infty}\leq O(m^{-15/24}),$$

the following holds:

$$\|\nabla(g,t)-\nabla(f,t)\|_{2,1}\leq O(NJm^{13/24}).$$

The randomness is because $a(\tau)\in\mathbb{R}^m$, $U(\tau)\in\mathbb{R}^{d\times m}$, $b(\tau)\in\mathbb{R}^m$, for $\tau=0$.

Proof. Notice that $\nabla(g,t) = \nabla_U \mathcal{L}(g_{U(t)}, \mathcal{S}(t))$ and $\nabla(f,t) = \nabla_U \mathcal{L}(f_{U(t)}, \mathcal{S}(t))$. By Claim E.6, with the given probability we have

$$\sum_{r=1}^{m} \mathbb{1}\{\nabla_r(g,t) \neq \nabla_r(f,t)\} \leq O(NJm^{7/8}).$$

For indices $r \in [m]$ satisfying $\nabla_r(g,t) \neq \nabla_r(f,t)$, the following holds:

$$\begin{aligned}
\|\nabla_r(g,t) - \nabla_r(f,t)\|_2 &= \|\nabla_{U,r}\mathcal{L}(g_{U(t)}, S(t)) - \nabla_{U,r}\mathcal{L}(f_{U(t)}, S(t))\|_2 \\
&\leq |a_r| \cdot \frac{1}{NJ} \cdot \sum_{c=1}^{N}\sum_{j=1}^{J} \|x_{c,j}\|_2 \cdot \big| \mathbb{1}\{\langle \widetilde{U}_r, x_{c,j}\rangle + b_r \geq 0\} \\
&\quad - \mathbb{1}\{\langle U_r, x_{c,j}\rangle + b_r \geq 0\}\big| \\
&\leq \frac{1}{m^{1/3}} \cdot \frac{1}{NJ} \cdot \sum_{c=1}^{N}\sum_{j=1}^{J} \big| \mathbb{1}\{\langle \widetilde{U}_r, x_{c,j}\rangle + b_r \geq 0\} - \mathbb{1}\{\langle U_r, x_{c,j}\rangle + b_r \geq 0\}\big| \\
&\leq \frac{1}{m^{1/3}}.
\end{aligned}$$

where the first step is definition, the second step follows that the loss function is 1-Lipschitz, the third step follows from $|a_r| \leq \frac{1}{m^{1/3}}$ and $\|x_{c,j}\|_2 = 1$, the last step follows from the bound of the indicator function. Thus, we do the conclusion:

$$\begin{aligned}
&\|\nabla(g,t) - \nabla(f,t)\|_{2,1} \\
=& \sum_{r=1}^{m} \|\nabla_r(g,t) - \nabla_r(f,t)\|_2 \cdot \mathbb{1}\{\nabla_r(g,t) \neq \nabla_r(f,t)\} \\
\leq& \frac{1}{m^{1/3}} \sum_{r=1}^{m} \mathbb{1}\{\nabla_r(g,t) \neq \nabla_r(f,t)\} \\
\leq& \frac{1}{m^{1/3}} \cdot O(NJm^{7/8}) \\
=& O(NJm^{13/24})
\end{aligned}$$

and finish the proof. □

E.5. Bounding Auxiliary

Claim E.6 (Bounding auxiliary)**.** *Let $\rho := \exp(-\Omega(m^{1/3}))$. With probability $\geq 1 - \rho$, we have*

$$\sum_{r=1}^{m} \mathbb{1}\{\nabla_r(g,t) \neq \nabla_r(f,t)\} \leq O(NJm^{7/8}).$$

The randomness is from $a(\tau) \in \mathbb{R}^m$, $U(\tau) \in \mathbb{R}^{d\times m}$, $b(\tau) \in \mathbb{R}^m$ for $\tau = 0$.

Proof. For $r \in [m]$, let $I_r := \mathbb{1}\{\nabla_r(g,t) \neq \nabla_r(f,t)\}$. By Claim A.4 we know that for each $x_{c,j}$ we have

$$\Pr[|\langle \widetilde{W}_{c,r}, x_{c,j}\rangle + b_r| \leq m^{-15/24}] \leq O(m^{-1/8}).$$

By putting a union bound on c and j, we get

$$\Pr\left[\exists c \in [N], j \in [J],\ |\langle \widetilde{W}_{c,r}, x_{c,j}\rangle + b_r| \leq m^{-15/24}\right] \leq O(NJm^{-1/8}).$$

Since

$$\Pr[I_r = 1] \leq \Pr[\exists j \in [J], c \in [N],\ |\langle \widetilde{W}_{c,r}, x_{c,j}\rangle + b_r| \leq m^{-15/24}],$$

we have

$$\Pr[I_r = 1] \leq O(NJm^{-1/8}).$$

By applying concentration inequality on I_r (independent Bernoulli) for $r \in [m]$, we obtain that with prob.

$$\geq 1 - \exp(-\Omega(NJm^{7/8})) > 1 - \rho,$$

the following holds:

$$\sum_{r=1}^{m} I_r \leq O(NJm^{7/8}).$$

Thus we finish the proof.

□

E.6. Further Discussion

Note that in the proof of Theorem E.3 we set the hidden layer's width m to be greater than $O(\epsilon^{-12})$, which seems impractical in reality: if we choose our convergence accuracy to be 10^{-2}, the width will become 10^{24} which is impossible to achieve.

However, we want to claim that the "-12" term is not intrinsic in our theorem and proof, and we can actually further improve the lower bound of m to $O((R/\epsilon)^{c_2})$ where c_2 is some constant between -3 and -4. To be specific, we observe from Eq. (11) that the "-12" term comes from $\frac{2}{3} - \frac{3}{4} = -\frac{1}{12}$, where $\frac{2}{3}$ appears in Lemma E.4 and $\frac{3}{4}$ appears in the assumption that $D_{U^*} \leq R/m^{3/4}$ in Definition E.2. As for our observations, the $\frac{2}{3}$ term is hard to improve. On the other hand, we can actually adjust the value of D_{U^*} as long as we ensure

$$D_{U^*} \leq R/m^{c_3}$$

for some constant $c_3 \in (0, 1)$. When we let $c_3 \to 1$, the final result will achieve

$$O((R/\epsilon)^3)$$

which is much more feasible in reality.

As the first work and the first step towards understanding the convergence of federated adversarial learning, the priority of our work is not achieving the tightest bounds. Instead, our main goal is to show the convergence of a general federated adversarial learning framework. We leave the task of improving the bound as future work.

F. Existence

In this section we prove the existence of U^* that is close to $U(0)$ and makes $\mathcal{L}_{\mathcal{A}^*}(f_{U^*})$ close to zero.

F.1. Tools from Previous Work

In order to prove our existence result, we first state two lemmas that will be used.

Lemma F.1 (Lemma 6.2 from (Zhang et al., 2020b)). *Suppose that $\|x_{c_1,j_1} - x_{c_2,j_2}\|_2 \geq \delta$ holds for each pair of two different data points x_{c_1,j_1}, x_{c_2,j_2}. Let $D = 24\gamma^{-1}\ln(48NJ/\epsilon)$, then there $\exists$ a polynomial $g : \mathbb{R} \to \mathbb{R}$ with size of coefficients no bigger than $O(\gamma^{-1}2^{6D})$ and degree no bigger than D, that satisfies for all $c_0 \in [N], j_0 \in [J]$ and $\widetilde{x}_{c_0,j_0} \in \mathcal{B}_2(x_{c_0,j_0}, \rho)$,*

$$\left| \sum_{c=1}^{N} \sum_{j=1}^{J} y_{c,j} \cdot g(\langle x_{c,j}, \widetilde{x}_{c_0,j_0} \rangle) - y_{c_0,j_0} \right| \leq \frac{\epsilon}{3}.$$

We let $f^*(x) := \sum_{c=1}^N \sum_{j=1}^J y_{c,j} \cdot g(\langle x_{c,j}, x\rangle)$ and have $|f^*(\widetilde{x}_{c_0,j_0}) - y_{c_0,j_0}| \leq \epsilon/3$.

Lemma F.2 (Lemma 6.5 from (Zhang et al., 2020b))**.** *Suppose* $\epsilon \in (0,1)$. *Suppose*

$$M = \mathrm{poly}((NJ/\epsilon)^{1/\gamma}, d) \quad \textit{and} \quad R = \mathrm{poly}((NJ/\epsilon)^{1/\gamma})$$

As long as $m \geq M$, *with prob.* $\geq 1 - \exp(-\Omega(\sqrt{m/NJ}))$, *there* $\exists U^* \in \mathbb{R}^{d\times m}$ *that satisfies*

$$\|U^* - U(0)\|_{2,\infty} \leq R/m^{2/3} \quad \textit{and} \quad \sup_{x\in\mathcal{X}} |g_{U^*}(x) - f^*(x)| \leq \epsilon/3.$$

The randomness is due to $a(\tau) \in \mathbb{R}^m$, $U(\tau) \in \mathbb{R}^{d\times m}$, $b(\tau) \in \mathbb{R}^m$ *for* $\tau = 0$.

F.2. Existence Result

We are going to present Theorem F.3 in this section and present its proofs.

Theorem F.3 (Existence, formal version of Theorem 5.2)**.** *Suppose that* $\epsilon \in (0,1)$. *Suppose*

$$M_0 = \mathrm{poly}(d, (NJ/\epsilon)^{1/\gamma}) \quad \textit{and} \quad R = \mathrm{poly}((NJ/\epsilon)^{1/\gamma})$$

As long as $m \geq M_0$, *then with prob.* $\geq 1 - \exp(-\Omega(m^{1/3}))$, *there exists* $U^* \in \mathbb{R}^{d\times m}$ *satisfying*

$$\|U^* - U(0)\|_{2,\infty} \leq R/m^{2/3} \quad \textit{and} \quad \mathcal{L}_{\mathcal{A}^*}(f_{U^*}) \leq \epsilon.$$

The randomness comes from $a(\tau) \in \mathbb{R}^m$, $U(\tau) \in \mathbb{R}^{d\times m}$, $b(\tau) \in \mathbb{R}^m$ *for* $\tau = 0$.

Proof. For convenient, we define

$$\rho_0 := \exp(-\Omega(\sqrt{m/NJ})) - \exp(-\Omega(m^{1/3})).$$

From Lemma F.1 we obtain the function f^*. From Lemma F.2 we know the existence of $M_0 = \mathrm{poly}(d, (NJ/\epsilon)^{1/\gamma})$ and also $R = \mathrm{poly}((NJ/\epsilon)^{1/\gamma})$.

By combining these two results with Theorem D.2, we have that for all $m \geq \mathrm{poly}(d, (NJ/\epsilon)^{1/\gamma})$, with prob.

$$\geq 1 - \rho_0,$$

there $\exists U^* \in \mathbb{R}^{d\times m}$ that satisfies $\|U^* - U(0)\|_{2,\infty} \leq R/m^{2/3}$.

In addition, the following properties:

- $\max_{x\in\mathcal{X}} |g_{U^*}(x) - f^*(x)|$ is at most $\epsilon/3$
- $\max_{x\in\mathcal{X}} |f_{U^*}(x) - g_{U^*}(x)|$ is at most $O(R^2/m^{1/6})$

Consider the loss function. For all $c \in [N]$, $j \in [J]$ and $\widetilde{x}_{c,j} \in \mathcal{B}(x_{c,j}, \rho)$, we have

$$\begin{aligned}
\ell(f_{U^*}(\widetilde{x}_{c,j}), y_{c,j}) &\leq |f_{U^*}(\widetilde{x}_{c,j}) - y_{c,j}| \\
&\leq |f_{U^*}(\widetilde{x}_{c,j}) - g_{U^*}(\widetilde{x}_{c,j})| + |g_{U^*}(\widetilde{x}_{c,j}) - f^*(\widetilde{x}_{c,j})| + |f^*(\widetilde{x}_{c,j}) - y_{c,j}| \\
&\leq O(R^2/m^{1/6}) + \frac{\epsilon}{3} + \frac{\epsilon}{3} \\
&\leq \epsilon,
\end{aligned}$$

Thus, we have that

$$\mathcal{L}_{\mathcal{A}^*}(f_{U^*}) = \frac{1}{NJ}\sum_{c=1}^N \sum_{j=1}^J \max \ell\left(f_{U^*}(x^*_{c,j}), y_{c,j}\right) \leq \epsilon.$$

Furthermore, since the m we consider satisfies $m \geq \Omega((NJ)^{1/\gamma})$, the holding probability is

$$\begin{aligned} &\geq 1 - \rho_0 \\ &= 1 - \exp(-\Omega(m^{1/3})). \end{aligned}$$

Thus, it finishes the proof of this theorem.

□