VAFL: a Method of Vertical Asynchronous Federated Learning

Tianyi Chen[⋆] **Xiao Jin**[⋆] **Yuejiao Sun**[†] **Wotao Yin**[†]

[⋆] *Rensselaer Polytechnic Institute - Troy, NY 12180, USA*
[†] *University of California - Los Angeles, Los Angeles, CA 90095, USA*
{CHENT18,JINX2}@RPI.EDU {SUNYJ,WOTAOYIN}@MATH.UCLA.EDU

Abstract

Federated learning (FL) enables model training with multi-client data where privacy concerns and coordination challenges prevent us from using conventional machine learning methods. In particular, horizontal FL handles multi-client data that share the same set of features, and vertical FL handles data of the same set of subjects but different clients hold their different features. This paper formulates vertical FL as an optimization problem, and develops a *simple yet practical* FL method. The new method allows each client to run stochastic gradient algorithms without coordination with other clients, so it is suitable for intermittent, even uncontrollable clients. This method further uses a new technique of *perturbed local embedding* to ensure data privacy and improve communication efficiency. Theoretically, we present the convergence rate and differential-privacy level of our method for strongly convex, nonconvex and even nonsmooth objectives separately. Empirically, we apply our method to federated logistic regression and deep learning on various image and healthcare datasets. The results compare favorably to centralized and synchronous FL methods.

1. Introduction

Federated learning (FL) is an emerging machine learning framework where a central server and multiple clients (e.g., mobile devices or organizations) collaboratively train a machine learning model (Konečný et al., 2016a; McMahan et al., 2017). Compared with existing distributed learning paradigms, FL raises new challenges including the difficulty of synchronizing multiple clients, the heterogeneity of data, and the privacy and security of both data and models. In most recent FL approaches, clients update local models and share them (as opposed to data or stochastic gradients) with a centralized server, and the server aggregate the models (McMahan et al., 2017). In this way, the server has no (direct) access to clients' data and training processes (Bonawitz et al., 2017).

Most of existing FL methods consider the scenario where each client has data of a different set of subjects but their data share many common features. Therefore, they can collaboratively learn a joint mapping from the feature space to the label space. This setting is also referred to data-partitioned or horizontal FL (Konečný et al., 2016b; McMahan et al., 2017).

Unlike the data-partitioned setting, in many learning scenarios, multiple clients handle data about the same set of subjects, but each client has a unique set of features. This case arises in e-commerce, financial, and healthcare applications (Hardy et al., 2017). For example, an e-commerce company may want to predict a customer's credit using her/his historical transactions from multiple financial institutions; and, a healthcare company wants to evaluate the health condition of a particular patient using his/her clinical data from various hospitals (Sun et al., 2019). In these examples, data owners (e.g., financial institutions and hospitals) have different records of those users in their joint user base, so by combining their features, they can establish a more accurate model. We refer to this setting as feature-partitioned or vertical FL (Yang et al., 2019).

Compared to the relatively well-studied horizontal FL setting, the vertical FL setting has its unique features and challenges. In horizontal FL, the global model update at a server is an additive aggregation of the local models, which are updated by each client using its own data. In contrast, the global model

Authors are listed in alphabetical order.

in vertical FL is the concatenation of local models, which are coupled by the loss function, so updating a client's local model requires the information of the other clients' models. Stronger model dependence in the vertical setting leads to challenges on privacy protection and communication efficiency.

1.1 This work

The present paper puts forth an optimization method for vertical FL, which is featured by three main components.

1. A *general optimization formulation* for vertical FL that consists of a global model and one local embedding model for each client. The local embedding model can be linear or nonlinear, or even nonsmooth. It maps raw data to compact features and, thus, reduces the number of parameters that need to be communicated to and from the global model.
2. Flexible *federated learning algorithms* that allow intermittent or even strategic client participation, uncoordinated training data selections, and data protection by differential-privacy based methods (for specific loss functions, one can instead apply multiple-party secure computing protocols).
3. *Rigorous analysis* that establishes the performance lower bound and the privacy level.

From an optimization-designer's perspective, to the best of our knowledge, we found our method to be the first that lets multiple clients upload their local models using uncoordinated SGD samples. The method also aggregates local model updates in an asynchronous, block-coordinate gradient fashion. Because of these features, this method and its convergence analysis are new and of their own interests to the distributed optimization community.

We have also numerically validated our vertical FL algorithms and their analyses on federated logistic regression and deep learning. Tests on image and medical datasets demonstrate the competitive performance of our algorithms relative to centralized and synchronous FL algorithms.

1.2 Prior art

Federated learning. Since the seminal work (Konečnỳ et al., 2016a; McMahan et al., 2017), there has been a large body of studies on FL in diverse settings. The most common FL setting is the horizontal setting, where a large set of data are partitioned among clients that share the same feature space (Konečnỳ et al., 2016b). To account for the personalization, multi-task FL has been studied in (Smith et al., 2017) that preserves the specialty of each client while also leveraging the similarity among clients, and horizontal FL with local representation learning has been empirically studied in (Liang et al., 2020). Agnostic FL has also been proposed in (Mohri et al., 2019), where the federated model is optimized for any target distribution formed by a mixture of the client distributions. Communication efficiency has been an important issue in FL. Popular methods generally aim to: c1) reduce the number of bits per communication round, including (Seide et al., 2014; Alistarh et al., 2017; Strom, 2015; Aji and Heafield, 2017; Stich et al., 2018), to list a few; and, c2) save the number of communication rounds (Chen et al., 2018; Wang and Joshi, 2018; Yu et al., 2019).

Privacy-preserving learning. More recently, feature-partitioned vertical FL has gained popularity in the financial and healthcare applications (Hardy et al., 2017; Yang et al., 2019; Niu et al., 2019; Kairouz et al., 2019). Different from the aggregated gradients in the horizontal case, the local gradients in the vertical FL may involve raw data of those features owned by other clients, which raises additional concerns on privacy. Data privacy has been an important topic since decades ago (Yao, 1982; Sweeney, 2002). But early approaches typically require expensive communication and signaling overhead when they are applied to the FL settings. Recently, the notion of differential privacy becomes popular because i) it is a quantifiable measure of privacy (Dwork et al., 2014; Abadi et al., 2016; Dong et al., 2019); and, ii) many existing learning algorithms can achieve differential privacy via simple modifications. In the context of learning from multiple clients, it has been studied in (Bonawitz et al., 2017; Hamm et al., 2016). But all these approaches are not designed for the vertical FL models and the flexible client update protocols.

Asynchronous and parallel optimization. Regarding methodology, asynchronous and parallel optimization methods are often used to solve problems with asynchrony and delays, e.g., (Recht et al., 2011). For the feature-partitioned vertical FL setting in this paper, it is particularly related to the Block Coordinate Descent (BCD) method (Xu and Yin, 2013; Razaviyayn et al., 2013). The asynchronous BCD and its stochastic variant have been developed under the condition of bounded delay in (Peng et al., 2016; Lian et al., 2017; Cannelli et al., 2016). The Recent advances in this direction aim to establish convergence under unbounded delay (Sun et al., 2017; Hannah and Yin, 2018). However, all these methods consider the shared memory structure that significantly reduces the negative effect of asynchrony and delays. Moreover, the state-of-the-art asynchronous methods cannot guarantee i) the convergence when the loss function is nonsmooth, and, ii) the privacy of the local update which is at the epicenter of the FL paradigm.

2. Vertical federated learning

In this section, we introduce the optimization formulation of vertical FL, and present our private and asynchronous client update for solving this problem.

2.1 Problem statement

Consider a set of M clients: $\mathcal{M} := \{1, \ldots, M\}$. A dataset of N samples, $\{\mathbf{x}_n, y_n\}_{n=1}^N$, are maintained by M local clients, where each client m maintains $x_{n,m} \in \mathbb{R}^{p_m}$ for $n = 1, \ldots, N$. Vector $x_{n,m}$ is also the m-th block of sample vector $\mathbf{x}_n := [x_{n,1}^\top, \cdots, x_{n,M}^\top]^\top$ at client m. Suppose the n-th label y_n is stored at the server.

To preserve the privacy of data, the client data $x_{n,m} \in \mathbb{R}^{p_m}$ are not shared with other clients as well as the server. Instead, each client m learns a local (linear or nonlinear, possibly nonsmooth) embedding h_m parameterized by θ_m that maps the high-dimensional vector $x_{n,m} \in \mathbb{R}^{p_m}$ to a low-dimensional one $h_{n,m} := h_m(\theta_m; x_{n,m}) \in \mathbb{R}^{\underline{p_m}}$ with $\underline{p_m} \ll p_m$. Ideally, the clients and the server want to collaboratively solve the following problem

$$F(\theta_0, \boldsymbol{\theta}) := \frac{1}{N} \sum_{n=1}^{N} \ell\left(\theta_0, h_{n,1}, \ldots, h_{n,M}; y_n\right) + \sum_{m=1}^{M} r(\theta_m)$$
$$\text{with} \quad h_{n,m} := h_m(\theta_m; x_{n,m}), \quad m = 1, \cdots, M \tag{1}$$

where θ_0 is the global model parameter kept at and learned by the server, and $\boldsymbol{\theta} := [\theta_1^\top, \cdots, \theta_M^\top]^\top$ concatenates the local models kept at and learned by local clients, ℓ is the loss capturing the accuracy of the global model parameters $\theta_0, \theta_1, \ldots, \theta_M$, and r is the per-client regularizer that confines the complexity of or encodes the prior knowledge about the local model parameters.

For problem (1), the local information of client m is fully captured in the embedding vector $h_{n,m}$, $\forall n = 1, \cdots, N$. Hence, the quantities that will be exchanged between server and clients are $\{h_{n,m}\}$ and the gradients of $\ell(\theta_0, h_{n,1}, \ldots, h_{n,M}; y_n)$ with respect to (w.r.t.) $\{h_{n,m}\}$. See a diagram for VAFL implementation in Figure 1.

A standard optimization method to solve (1) is the (synchronous) block-wise stochastic gradient method (Xu and Yin, 2015). Specifically, at iteration k, the server performs

$$\theta_0^{k+1} = \theta_0^k - \frac{\eta_0^k}{N_k} \sum_{n \in \mathcal{N}_k} \nabla_{\theta_0} \ell(\theta_0^k, h_{n,1}^k, \ldots, h_{n,M}^k; y_n) \tag{2}$$

where $h_{n,m}^k := h_m(\theta_m^k; x_{n,m})$, $\mathcal{N}_k$ is a minibatch randomly selected at iteration k with the sample size $N_k = |\mathcal{N}_k|$, and η_0^k is the stepsize at iteration k. With the *same* minibatch of data $\mathcal{N}_k$, the model

Algorithm 1 Vertical asynchronous federated learning

```
 1: initialize: $\theta_0$, $\{\theta_m\}$, datum index $n$, client index $m$
 2: while not convergent do
 3:   when a Client $m$ is activated, do:
 4:     †select private datum (or data mini-batch) $x_{n,m}$
 5:     †upload secure information $h_{n,m} = h_m(\theta_m; x_{n,m})$
 6:     query $\nabla_{h_{n,m}} \ell(\theta_0, h_{n,1}, \ldots, h_{n,M}; y_n)$ from Server
 7:     update local model $\theta_m$
 8:   when Server receives $h_{n,m}$ from Client $m$, do:
 9:     compute $\nabla_{\theta_0} \ell(\theta_0, h_{n,1}, \ldots, h_{n,M}; y_n)$
10:     update server's local model $\theta_0$
11:   when Server receives a query from Client $m$, do:
12:     compute $\nabla_{h_{n,m}} \ell(\theta_0, h_{n,1}, \ldots, h_{n,M}; y_n)$
13:     send it to Client $m$
14: end while
```

†We can let Step 5 also send $h_{n,m}$ for those n *not* selected in Step 4. We can re-order Steps 4–7 as 6, 7, *then* 4, and 5. They reduce information delay, yet analysis is unchanged.

Algorithm 2 Vertical t-synchronous federated learning

```
 1: Initialize: $\theta_0$, $\{\theta_m\}$, datum index $n$, client index $m$, integer $1 \le t \le M$
 2: while not convergent do
      Algorithm 1, Lines 3–7
 8:   when Server receives $h_{n,m}$'s from $t$ Clients, do:
        Algorithm 1, Lines 9 and 10
11:   when Server receives queries from $t$ Clients, do:
        Algorithm 1, Lines 12 and 13 for each of $t$ clients
14: end while
```

update at each local client m is

$$\theta_m^{k+1} = \theta_m^k - \frac{\eta_m^k}{N_k} \sum_{n \in \mathcal{N}_k} \nabla_{\theta_m} h_{n,m}^k \nabla_{h_{n,m}} \ell(\theta_0^k, h_{n,1}^k, \ldots, h_{n,M}^k; y_n) - \eta_m^k \nabla r(\theta_m^k) \tag{3}$$

where η_m^k is the stepsize at iteration k. When r is nonsmooth and a proximal mapping, we can replace $-\eta_m^k \nabla r(\theta_m^k)$ by an extra proximal step.

Implementing updates (2) and (3) requires all the clients to use the same minibatch of data to update their local model simultaneously. In FL, however, clients (e.g., devices) participate in the learning process only when they are available (plugged in, waked up, and/or connected with unmetered WiFi) and thus it is difficult to obtain the up-to-date information of all clients at any time. Moreover, the local embedding vectors and the local gradients exchanged between server and clients still have risk of privacy leakage.

2.2 Asynchronous client updates

For FL, we consider solving (1) without coordination among clients. Asynchronous optimization methods have been used to solve such problems. However, state-of-the-art methods cannot guarantee i) the convergence when the mapping $h_{n,m}$ is nonlinear (thus the loss is nonsmooth), and, ii) the privacy of the update which is at the epicenter of the FL paradigm.

In a high level, we describe our vertical asynchronous federated learning (VAFL) algorithm as follows. During the learning process, from the server side, it waits until receiving a message from an

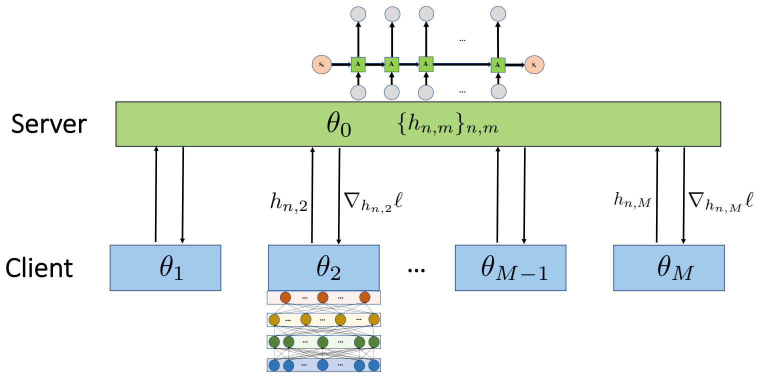

Figure 1: A diagram for VAFL implementation.

active client m, which is either
i) **a query** of the loss function's gradient w.r.t. to the embedding vector $h_{n,m}$; or,
ii) **a new embedding vector** $h_{n,m}$ calculated using the updated local model parameter θ_m.

To response to the query i), the server calculates the gradient for client m using its current $\{h_{n,m}\}$, and sends it to the client; and, upon receiving ii), the server computes the new gradient w.r.t. θ_0 using the embedding vectors it currently has from other clients and updates its model θ_0.

For each interaction with server, each *active* client m randomly selects a datum $x_{n,m}$, queries the corresponding gradient w.r.t. $h_{n,m}$ from server, and then it securely uploads the updated embedding vector $h_{n,m}$, and then updates the local model θ_m. The mechanism that ensures secure uploading will be described in Section 4. Without introducing cumbersome iteration, client, and sample indexes, we summarize the asynchronous client updates in Algorithm 1.

Specifically, since clients update the model without external coordination, we thereafter use k to denote the global counter (or iteration), which increases by one whenever i) the server receives the new embedding vector $h_{n,m}$ from a client, calculates the gradient, and updates the server model θ_0; and, ii) the corresponding client m obtains the gradient w.r.t. $h_{n,m}$, and updates the local model θ_m. Accordingly, we let m_k denote the client index that uploads at iteration k, and n_k denote the sample index used at iteration k.

For notation brevity, we use a single datum n_k for each uncoordinated update in the subsequent algorithms, but the algorithm and its analysis can be easily generalized to a minibatch of data $\mathcal{N}_k$. Let $\hat{g}_0^k$ denote the stochastic gradients of the loss at n_k-th sample w.r.t. server model θ_0 as

$$\hat{g}_0^k := \nabla_{\theta_0}\ell\left(\theta_0^k, h_{n_k,1}^{k-\tau_{n_k,1}^k}, \ldots, h_{n_k,M}^{k-\tau_{n_k,M}^k}; y_{n_k}\right) \tag{4a}$$

and the gradients w.r.t. the local model θ_m as

$$\begin{aligned}\hat{g}_m^k :=& \nabla_{\theta_m}\ell\left(\theta_0^k, h_{n_k,1}^{k-\tau_{n_k,1}^k}, \ldots, h_{n_k,M}^{k-\tau_{n_k,M}^k}; y_{n_k}\right) \\ =& \nabla_{\theta_m} h_{n_k,m}^k \nabla_{h_{n_k,m}}\ell\left(\theta_0^k, h_{n_k,1}^{k-\tau_{n_k,1}^k}, \ldots, h_{n_k,M}^{k-\tau_{n_k,M}^k}; y_{n_k}\right).\end{aligned} \tag{4b}$$

The delay for client m and sample n will increase via

$$\tau_{n,m}^{k+1} = \begin{cases} 1, & m = m^k,\ n = n^k, \\ \tau_{n,m}^k + 1, & \text{otherwise.} \end{cases} \tag{5}$$

With the above short-hand notation, at iteration k, the update at the **server side** is

$$\theta_0^{k+1} = \theta_0^k - \eta_0^k \hat{g}_0^k. \tag{6}$$

For the **active local client** m_k at iteration k, its update is

$$\theta_{m_k}^{k+1} = \theta_{m_k}^k - \eta_{m_k}^k g_{m_k}^k - \eta_{m_k}^k \nabla r(\theta_{m_k}^k), \tag{7a}$$

and for the **other clients** $m \neq m_k$, the update is

$$\theta_m^{k+1} = \theta_m^k, \tag{7b}$$

where η_m^k is the stepsize and m_k is the index of the client responsible for the kth update.

2.3 Types of flexible update rules

As shown in (4), the stochastic gradients are evaluated using delayed local embedding information $h_m^{k-\tau_{n_k,m}^k}$ from each client m, where $\tau_{n_k,m}^k$ is caused by both asynchronous communication and stochastic sampling. Unexpected delays can cause the divergence of the algorithm.

To ensure convergence, we consider two reasonable settings on the flexible update protocols:
1. *Uniformly bounded delay D.* We can realize this by modifying the server behavior. During the training process, whenever the delay of $\tau_{n_k,m}^k$ exceeds $D(>0)$, the server immediately queries fresh $h_{n,m}$ from client m before continuing the server update process.
2. *Stochastic unbounded delay.* In this case, the activation of each client is a stochastic process. The delays is determined by the hitting times of the stochastic processes. For example, if the activation of all the clients follows independent Poisson processes, the delays will be geometrically distributed.
3. *t-synchronous update, $t > 0$.* While fully asynchronous update is most flexible, t-synchronous update is also commonly adopted. In this case, the server computes the gradient w.r.t. θ_0 until receiving $\{h_{n,m}\}$ from t different clients, and then updates the server's model using the newly computed gradient. The t-synchronous updates have more stable performance empirically, which is listed in Algorithm 2.

In this paper, we do not consider dishonest and malicious clients and server, but it is possible to incorporate robust learning techniques into our vertical FL algorithms based on recent works (Bhagoji et al., 2019; Chen et al., 2017).

3. Convergence analysis

We present the convergence results of our vertical asynchronous federated learning (VAFL) method for the nonconvex and strongly convex cases and under different update rules. Due to space limitation, this section mainly presents the convergence rates for fully asynchronous version of VAFL (Algorithm 1), and the convergence results for t-synchronous one (Algorithm 2) are similar, and thus will be given in the supplementary materials. The proofs of all the lemmas and theorems presented in this section can be also found in the supplementary materials.

To analyze the performance of Algorithm 1, we first make the following assumptions on sampling and smoothness.

Assumption 1 *Sample indexes $\{n_k\}$ are i.i.d. from $\{1, \ldots, N\}$ and $\mathbb{P}(n_k = n) = 1/N$. And the variance of gradient follows*

$$\mathbb{E}\left[\|g_m^k - \nabla_{\theta_m} F(\theta_0^k, \boldsymbol{\theta}^k)\|^2\right] \leq \sigma_m^2,\ m = 0, \ldots, M \tag{8}$$

where g_m^k is the stochastic gradient $\hat{g}_m^k$ without delay, e.g., $g_m^k := \nabla_{\theta_m} \ell\left(\theta_0^k, h_{n_k,1}^k, \ldots, h_{n_k,M}^k; y_{n_k}\right)$.

Assumption 2 *The optimal loss is lower bounded $F^* > -\infty$. The gradient ∇F is L-Lipschitz continuous, and $\nabla_{\theta_m} F$ is L_m-Lipschitz continuous.*

Assumption 1 is common in analyzing SGD (Bottou et al., 2016). Generally, assumption 2 cannot be satisfied under our general vertical FL formulation with *nonsmooth* local embedding functions such as neural networks. However, techniques that we call perturbed local embedding will be introduced to enforce smoothness in Section 4.

To handle asynchrony, we need the following assumption, which is often seen in the analysis of asynchronous BCD.

Assumption 3 *The uploading client index m_k is random and independent of $m_0, \ldots, m_{k-1}$ and satisfies*

$$\mathbb{P}(m_k = m) := q_m > 0. \tag{9}$$

A simple scenario satisfying this assumptions is that the activation of all clients follows independent Poisson processes. That is, if the time difference between two consecutive activations of client m follows exponential distribution with parameter λ_m, then the activation of client m is a Possion process with $q_m = \lambda_m^{-1} / \sum_{j=1}^{M} \lambda_j^{-1}$.

We first present the convergence results for bounded $\tau_{n_k,m}^k$.

3.1 Convergence under bounded delay

We make the following assumption *only* for this subsection.

Assumption 4 (Uniformly bounded delay) *For each client m and each sample n, the delay $\tau_{n,m}^k$ at iteration k is bounded by a constant D, i.e., $\tau_{n,m}^k \leq D$.*

To handle the delayed information, we leverage the following Lyapunov function for analyzing VAFL

$$V^k := F^k + \sum_{d=1}^{D} \gamma_d \|\boldsymbol{\theta}^{k-d+1} - \boldsymbol{\theta}^{k-d}\|^2 \tag{10}$$

where $\{\gamma_d\}$ are a set of constants to be determined later.

Lemma 1 *Under Assumptions 1 – 4, for $\eta_0^k \leq \frac{1}{4L}, \eta_m^k \leq \frac{1}{4(L+2\gamma_1)}$, it follows that (with $\gamma_{D+1} = 0$)*

$$\begin{aligned}
\mathbb{E}V^{k+1} - \mathbb{E}V^k \leq & -\left(\frac{\eta_0^k}{2} - L(\eta_0^k)^2\right) \mathbb{E}[\|\nabla_{\theta_0} F(\theta_0^k, \boldsymbol{\theta}^k)\|^2] \\
& - \sum_{m=1}^{M} q_m \left(\frac{\eta_m^k}{2} - L(\eta_m^k)^2 - 2\gamma_1 (\eta_m^k)^2\right) \mathbb{E}[\|\nabla_{\theta_m} F(\theta_0^k, \boldsymbol{\theta}^k)\|^2] \\
& + \sum_{d=1}^{D} \left(Dc^k + D\gamma_1 \max_m 2(\eta_m^k)^2 L_m^2 + \gamma_{d+1} - \gamma_d\right) \mathbb{E}[\|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\|^2] \\
& + L(\eta_0^k)^2 \sigma_0^2 + \sum_{m=1}^{M} q_m (L + 2\gamma_1)(\eta_m^k)^2 \sigma_m^2.
\end{aligned} \tag{11}$$

If $\{\gamma_d\}$ are chosen properly as specified in the supplementary materials, the first three terms in the right hand side of (11) is negative. By carefully choosing $\{\eta_0^k, \eta_m^k\}$, we can ensure the convergence of Algorithm 1.

We first present the convergence for the nonconvex case.

Theorem 2 *Under the same assumptions of Lemma 1, if $\eta_0^k = \eta_m^k = \min\{\frac{1}{4(1+D)L}, \frac{c_\eta}{\sqrt{K}}\}$ with $c_\eta > 0$, then we have*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\|\nabla F(\theta_0^k, \boldsymbol{\theta}^k)\|^2] = \mathcal{O}\left(1/\sqrt{K}\right). \tag{12}$$

Under the additional assumption of strong convexity, the convergence rate is improved.

Theorem 3 *Assume that F is μ-strongly convex in $(\theta_0, \boldsymbol{\theta})$. Then under the same assumptions of Lemma 1, if $\eta^k = \frac{4}{\mu \min\limits_m \sqrt{q_m}(k+K_0)}$ with $K_0 = \frac{4(4(D+1)L+\mu \min\limits_m \sqrt{q_m} D)}{\mu \min\limits_m \sqrt{q_m}}$, then*

$$\mathbb{E}F\left(\theta_0^K, \boldsymbol{\theta}^K\right) - F^* = \mathcal{O}\left(1/K\right). \tag{13}$$

It worth mentioning that under the assumption of bounded delay, without even coordinating clients' gradient samples and local model updates, our algorithm achieves the same order of convergence as that of block-wise SGD in both nonconvex and strongly convex cases (Xu and Yin, 2015).

We next present the convergence results for the case where $\tau_{n_k,m}^k$ is unbounded but stochastic.

3.2 Convergence under stochastic unbounded delay

We make the following assumption *only* for this subsection.

Assumption 5 (Stochastic unbounded delay) *For each client m, the delay $\tau_{n_k,m}^k$ is an random variable with unbounded support. And there exists $\bar{p}_m, \rho > 0$ such that*

$$\mathbb{P}\left(\tau_{n_k,m}^k = d\right) \le \bar{p}_m \rho^d := p_{m,d}. \tag{14}$$

Similar to (10), we define the following Lyapunov function for the stochastic unbounded delay setting

$$V^k = F^k + \sum_{m=1}^{M} \sum_{d=1}^{k} \gamma_{m,d} \|\theta_m^{k-d+1} - \theta_m^{k-d}\|^2 \tag{15}$$

where a set of constants $\{\gamma_{m,d}\}$ will be determined later. The Lyapunov function used in this stochastic and unbounded delay case is different from that in (10), since the delay distribution considered here is client-specific instead of a uniform bound in Section 3.1. Under Assumption 5, we obtain the convergence rates of the same order as those the under bounded delay assumption.

Theorem 4 *Under Assumptions 1-3 and 5, if $\eta_0^k = \eta_m^k = \min\left\{\frac{1}{4(1+\min_m \sqrt{c_m})L}, \frac{c_\eta}{\sqrt{K}}\right\}$, then we have*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\|\nabla F(\theta_0^k, \boldsymbol{\theta}^k)\|^2] = \mathcal{O}\left(1/\sqrt{K}\right). \tag{16}$$

Under the additional assumption of strong convexity, the convergence rate is improved.

Theorem 5 *Assume that F is μ-strongly convex in $(\theta_0, \boldsymbol{\theta})$. Then under Assumptions 1-3 and 5, if $\eta_0^k = \eta_m^k = \frac{2}{\nu(k+K_0)}$ where $K_0 = \frac{4(1+\max_m \sqrt{c_m})L}{\nu}$ and ν is a positive constant depending on $\mu, L, \bar{p}_m, \rho$, then it follows that*

$$\mathbb{E}F(\theta_0^K, \boldsymbol{\theta}^K) - F^* = \mathcal{O}\left(1/K\right). \tag{17}$$

Similar to Section 3.1, under the unbounded but stochastic delay, without even coordinating clients' gradient samples and local model updates, our algorithm achieves the same order of convergence as that of block-wise SGD in both nonconvex and strongly convex cases (Xu and Yin, 2015).

4. Perturbed local embedding: Enforcing differential privacy and smoothness

In this section, we introduce a local perturbation technique that is applied by each client to enforce the differential privacy of local information, which also smoothes the otherwise nonsmooth mapping of local embedding so the convergence analysis in Section 3 can be applied.

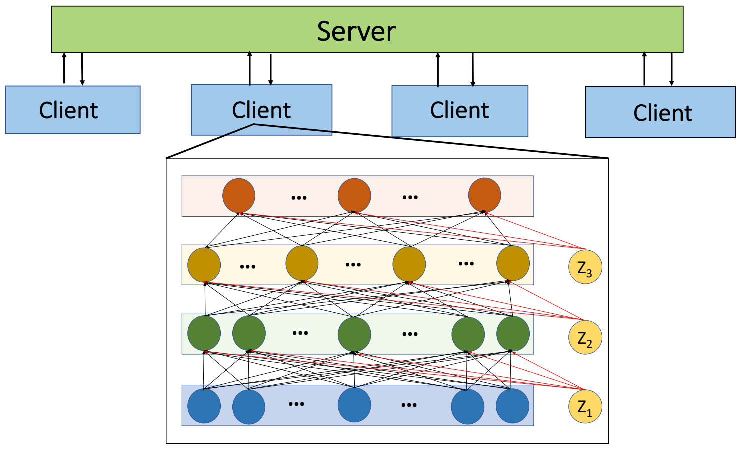

Figure 2: Perturbed local embedding implementation; Z_l is the random neuron added at layer l.

4.1 Local perturbation

Recall that h_m denotes a local embedding function of client m with the parameter θ_m which embeds the information of local data $x_{n,m}$ into its outputs $h_{n,m} := h_m(\theta_m; x_{n,m})$. When h_m is linear embedding, it is as simple as $h_m(\theta_m; x_{n,m}) = x_{n,m}^\top \theta_m$. To further account for nonlinear embedding such as neural networks, we represent $h_{n,m}$ in the following composite form

$$u_0 = x_{n,m} \tag{18a}$$
$$u_l = \sigma_l(w_l u_{l-1} + b_l), \quad l = 1, \ldots, L \tag{18b}$$
$$h_{n,m} = u_L \tag{18c}$$

where σ_l is a linear or nonlinear function, and w_l, b_l corresponds to the parameter θ_m of h_m, e.g., $\theta_m := [w_1, \cdots, w_L, b_1, \cdots, b_L]^\top$. Here we assume that σ_l is $L^0_{\sigma_l}$-Lipschitz continuous. Specially, when h_m is linear, the composite embedding (18) corresponds to $L = 1, \sigma_1(z) = z$.

We perturb the local embedding function h_m by adding a random neuron with output Z_l at each layer l (cf. (18b))

$$u_l = \sigma_l(w_l u_{l-1} + b_l + Z_l), \quad l = 1, \ldots, L \tag{19}$$

where $Z_1, \ldots, Z_L$ are independent random variables. With properly chosen distributions of $Z_l, l = 1, \ldots, L$, we show below h_m is smooth and enables differential privacy. While it does not exclude other options, our choice of the perturbation distributions is

$$Z_L \sim \mathcal{N}\left(0, c^2\right) \tag{20a}$$
$$Z_l \sim \mathcal{U}[-\sqrt{3}c_l, \sqrt{3}c_l], \quad l = 1, \cdots, L-1 \tag{20b}$$

where $\mathcal{N}\left(0, c^2\right)$ denotes the Gaussian distribution with zero mean and variance c^2, and $\mathcal{U}[-\sqrt{3}c_l, \sqrt{3}c_l]$ denotes the uniform distribution over $[-\sqrt{3}c_l, \sqrt{3}c_l]$.

4.2 Enforcing smoothness

We now connect the perturbed local embedding technique with the convergence theories in Section 3.

The convergence results in Section 3 hold under Assumption 2 which requires the smoothness of the overall loss function. Inspired by the randomized smoothing technique (Duchi et al., 2012; Nesterov and Spokoiny, 2017), we are able to smooth the objective function by taking expectation with respect

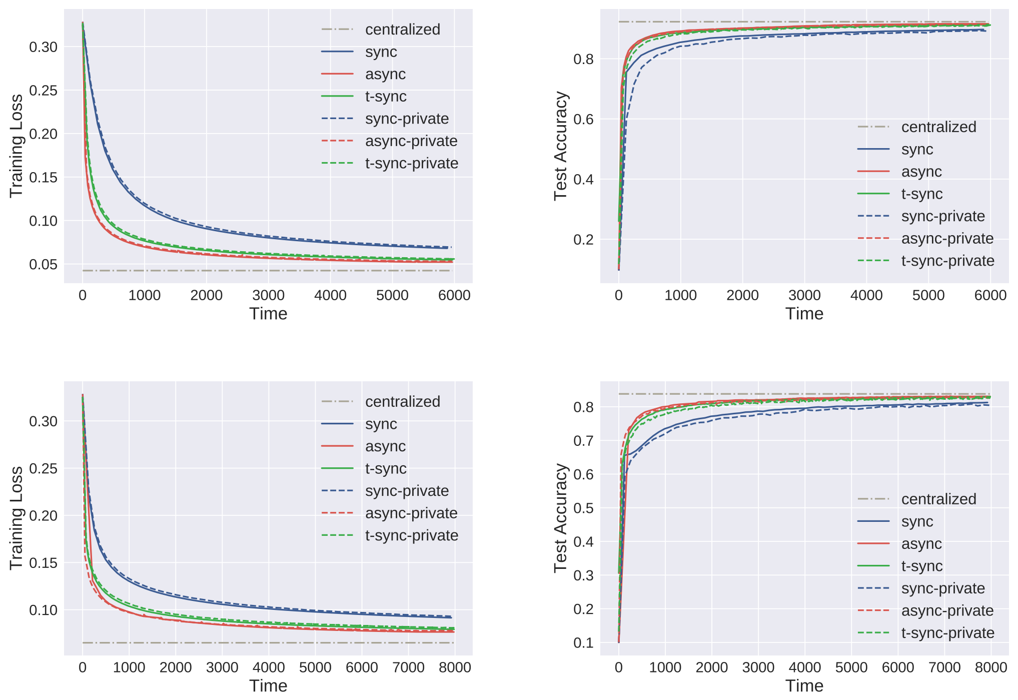

Figure 3: Training loss and testing accuracy versus clock time (sec) in MNIST (left two) and Fashion-MNIST (right two) datasets.

to the random neurons. Intuitively this follows the fact that the smoothness of a function can be increased by convolving with proper distributions. By adding random neuron Z_l, the landscape of σ_l will be smoothed in expectation with respect to Z_l. And by induction, we can show the smoothness of local embedding vector h_m. Then so long as the loss function ℓ is smooth w.r.t. the local embedding vector h_m, the global objective F is smooth by taking expectation with respect to all the random neurons.

We formally establish this result in the following theorem.

Theorem 6 *For each embedding function h_m, if the activation functions follow $\sigma_l = \sigma, \forall l$, and the variances of the random neurons follow (20), and assume $\|w_l\|$ is bounded, then with $\mathbf{Z} := [Z_1^\top, \cdots, Z_L^\top]^\top$, the perturbed loss satisfies Assumption 2, which is given by*

$$F_c(\theta_0, \boldsymbol{\theta}) := \mathbb{E}_{\mathbf{Z}}[F(\theta_0, \boldsymbol{\theta}; \mathbf{Z})]. \tag{21}$$

Starting from $L_{b_L}^h = L_\sigma^0 d/c$, the smoothness constants of the local model θ_m denoted as $L_{\theta_m}^{F_c}$ satisfy the following recursion ($l = 1, \cdots, L-1$)

$$\begin{aligned}
L_{b_l}^h &= L_{b_{l+1}}^h \|w_{l+1}\| (L_\sigma^0)^2 + L_\sigma^0 \|w_L\| \cdots L_\sigma^0 \|w_{l+1}\| L_{\bar{\sigma}}(c_l) \\
L_{w_l}^h &= \mathbb{E}[\|u_{l-1}\|] L_{b_l}^h \\
L_{\theta_m}^{F_c} &= L_{h_m}^\ell (L_{h_m}^0)^2 + L_\ell^0 \sum_{l=1}^{L} (L_{w_l}^h + L_{b_l}^h) + L_{\theta_m}^r
\end{aligned} \tag{22}$$

where $L_{\theta_m}^r$ is the smoothness constant of the regularizer w.r.t. θ_m; $L_{b_l}^h$ and $L_{w_l}^h$ are the smoothness constants of the perturbed local embedding h w.r.t. the bias b_l and weight w_l; and $L_{\bar{\sigma}}(c_l) := 2\sqrt{d} L_\sigma^0 / c_l$ is the smoothness constant of the neuron at lth layer under the uniform perturbation.

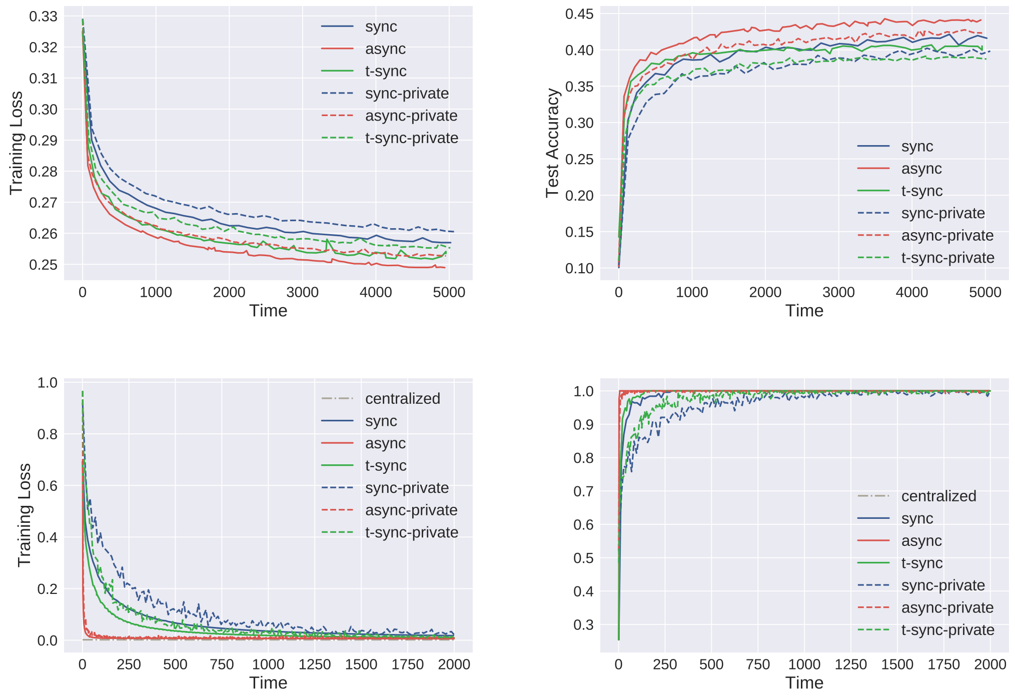

Figure 4: Training loss and testing accuracy versus clock time (sec) in CIFAR10 (left two) and Parkinson disease (right two) datasets.

Theorem 6 implies that the perturbed loss is smooth w.r.t. the local model θ_m, and a large perturbation (large c_l or c) will lead to a smaller smoothness constant.

Remark 7 *Note that even with the same model parameters, $F_c(\theta_0;\boldsymbol{\theta})$ in (21) is usually different from $F(\theta_0,\boldsymbol{\theta})$. However their difference $|F_c(\theta_0;\boldsymbol{\theta})-F(\theta_0,\boldsymbol{\theta})|$ are bounded and proportional to the standard deviation of the random neurons. See the appendix for the detailed estimation.*

4.3 Enforcing differential privacy

We now connect the perturbed local embedding technique with the private information exchange in Algorithms 1-2.

Privacy is a major concern in FL. As local clients keep sending out embedded information, it is essential to prevent any attacker to trace back to a specific individual via this observation. As a statistical tool that quantifies the privacy in the information sharing process, we introduce the recently developed Gaussian differential privacy (GDP) in (Dong et al., 2019).

Definition 8 ((Dong et al., 2019)) *A mechanism $\mathcal{M}$ is said to satisfy μ-GDP if for all neighboring datasets S and S', if*

$$T(\mathcal{M}(S),\mathcal{M}(S'))\geq T(\mathcal{N}(0,1),\mathcal{N}(\mu,1)) \tag{23}$$

where the trade-off function is defined as $T(P,Q)(\alpha)=\inf\{\beta_\phi:\alpha_\phi\leq\alpha\}$, and α_ϕ,β_ϕ are type I and II error rates given a threshold ϕ.

Intuitively, μ-GDP guarantees that distinguishing two adjacent datasets via information revealed by $\mathcal{M}$ is at least as difficult as distinguishing the two distributions $\mathcal{N}(0,1)$ and $\mathcal{N}(\mu,1)$. Smaller μ means less privacy loss.

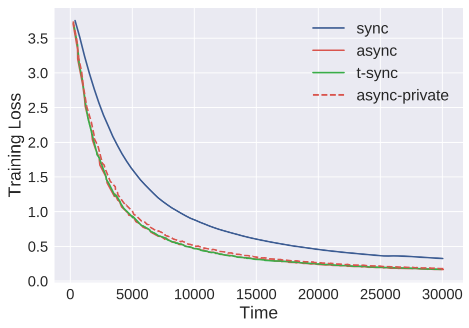

Figure 5: Training loss of VAFL with nonlinear local embedding on *ModelNet40* dataset.

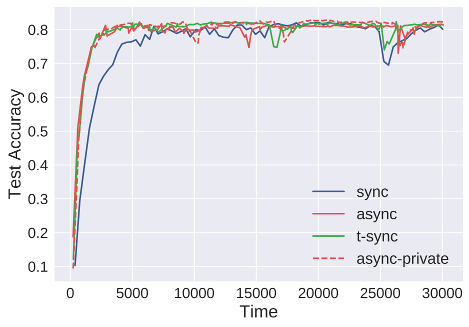

Figure 6: Testing accuracy of VAFL with nonlinear local embedding on *ModelNet40* dataset.

In our vertical FL framework, we will demonstrate that besides changing the nonsmooth landscape of the objective function, the random neurons added to the nonlinear embedding function can also guarantee the privacy of our vertical FL algorithms. To characterize the level of privacy of our local embedding approaches, we build on but go beyond the moments accountant technique originally developed in (Abadi et al., 2016) to account for our unique local composite embedding structure. By leveraging the composition and the subsampling theorems in (Dong et al., 2019), we next establish that adding random neurons endows Algorithm 1 with GDP.

Theorem 9 *Under the same set of assumptions as those in Theorem 6, for client m, if we set the variance of the Gaussian random neuron at the L-th layer as*

$$c = \mathcal{O}\left(N_m\sqrt{K}/(\mu N)\right) \tag{24}$$

where N_m is the size of minibatch used at client m, N is the size of the whole batch, K is the number of queries (i.e. the number of data samples processed by h_m at client m), then VAFL satisfies μ-GDP for the dataset of client m.

Remark 10 *Theorem 9 demonstrates the inherent trade-off between the learning accuracy and the level of privacy. To increase privacy, i.e., decrease μ in* (23)*, the variance of random neurons needs to be increased (cf.* (24)*). However, as the variance of random neurons increases, the variance of the stochastic gradient* (4) *also increases, which will in turn lead to slower convergence of Algorithm 1.*

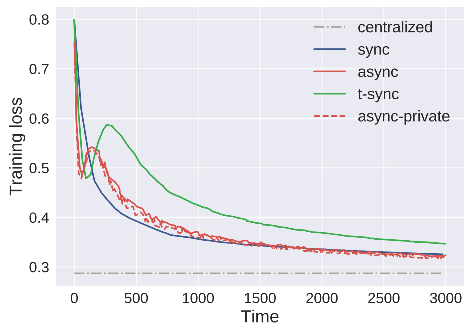

Figure 7: Training loss of VAFL with local LSTM embedding on *MIMIC-III* critical care dataset.

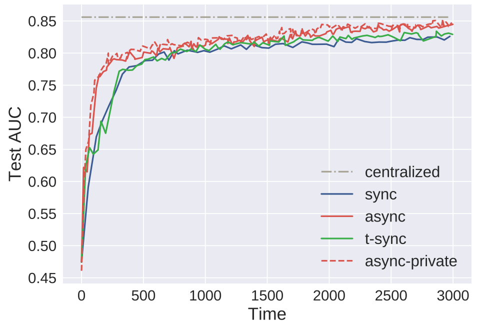

Figure 8: AUC curve of VAFL with local LSTM embedding on *MIMIC-III* clinical care dataset.

5. Numerical tests and remarks

To evaluate the performance of our proposed VAFL method, numerical tests have been conducted on both federated logistic regression and neural network models on multiple datasets. We benchmark the fully asynchronous version of VAFL (**async**) in Algorithm 1, and t-synchronous version of VAFL (**t-sync**) in Algorithm 2 with the synchronous block-wise SGD (**sync**), which requires synchronization and sample index coordination among clients in each iteration. We also include private versions of these algorithms via perturbed local embedding technique in Section 4. To simulate the client asynchrony, we use multiple threads as local clients and add random delay to each client. The random delay follows Poisson distribution with client-specific parameters to reflect heterogeneity. Details about the simulation environment, choice of parameters and data allocation among clients are presented in the supplement materials.

VAFL for federated logistic regression. We first conduct logistic regression on MNIST, Fashion-MNIST, CIFAR10 and Parkinson disease (Sakar et al., 2019) datasets. The l_2-regularizer coefficient is set as 0.001. We select $M = 7$ for MNIST and MNIST, $M = 8$ for CIFAR10 and $M = 3$ for PD dataset. The training loss and testing accuracy versus wall-clock time are reported in Figures 3 and 4. The dashed horizontal lines represent the results trained on the centralized (non-federated) model, and the dashed curves represent private variants of considered algorithms with variance $c = 0.1$. In all cases, the VAFL algorithms learn a federated model with accuracies comparable to that of the centralized model that requires collecting all raw data. And async and t-sync VAFL algorithms have two times speed-up in most cases.

VAFL for federated deep learning. We first train a neural network modified from MVCNN with 12-view data (Wu et al., 2015). We use $M = 4$ clients, and each client has 3 views of each object and use a 7-layer CNN as local embedding functions, and server uses a fully connected network to aggregate the local embedding vectors. Results are plotted in Figures 5 and 6. Centralized model in this multi-view learning case coincides with sync, and is omitted. Again async and t-sync have sizable speed-up relative to sync, and private async (with $c = 1$) does not sacrifice much in accuracy.

We also test our VAFL algorithm in MIMIC-III — an open dataset comprising deidentified health data (Johnson et al., 2016). We perform the in-hospital mortality prediction as in (Harutyunyan et al., 2019) among $M = 4$ clients. Each client uses LSTM as the embedding function. In Figures 7 and 8, we can still observe that async and t-sync VAFL learn a federated model with accuracies comparable to that of the centralized model, and requires less time relative to the synchronous FL algorithm.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proc. ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, Vienna, Austria, October 2016.

Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. In *Proc. Conf. Empirical Methods Natural Language Process.*, pages 440–445, Copenhagen, Denmark, Sep 2017.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Proc. Advances in Neural Info. Process. Syst.*, pages 1709–1720, Long Beach, CA, Dec 2017.

Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *Proc. Intl. Conf. Machine Learn.*, pages 634–643, Long Beach, CA, June 2019.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proc. ACM Conf. on Comp. and Comm. Security*, pages 1175–1191, Dallas, TX, October 2017.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *arXiv preprint:1606.04838*, June 2016.

Loris Cannelli, Francisco Facchinei, Vyacheslav Kungurtsev, and Gesualdo Scutari. Asynchronous parallel algorithms for nonconvex big-data optimization: Model and convergence. *arXiv preprint:1607.04818*, July 2016.

Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. LAG: Lazily aggregated gradient for communication-efficient distributed learning. In *Proc. Advances in Neural Info. Process. Syst.*, pages 5050–5060, Montreal, Canada, Dec 2018.

Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, pages 1–25, New York, NY, 2017.

Jinshuo Dong, Aaron Roth, and Weijie J Su. Gaussian differential privacy. *arXiv preprint:1905.02383*, May 2019.

John C Duchi, Peter L Bartlett, and Martin J Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Jihun Hamm, Yingjun Cao, and Mikhail Belkin. Learning privately from multiparty data. In *Proc. Intl. Conf. Machine Learn.*, pages 555–563, New York, NY, June 2016.

Robert Hannah and Wotao Yin. On unbounded delays in asynchronous parallel fixed-point algorithms. *Journal of Scientific Computing*, 76(1):299–326, July 2018.

Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint:1711.10677*, November 2017.

Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1):96, 2019. ISSN 2052-4463. doi: 10.1038/s41597-019-0103-9. URL `https://doi.org/10.1038/s41597-019-0103-9`.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(160035), 2016.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint:1912.04977*, December 2019.

Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint:1610.02527*, October 2016a.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint:1610.05492*, Oct 2016b.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In *Proc. Advances in Neural Info. Process. Syst.*, pages 5330–5340, Long Beach, CA, Dec 2017.

Paul Pu Liang, Terrance Liu, Liu Ziyin, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint:2001.01523*, January 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. Intl. Conf. Artificial Intell. and Stat.*, pages 1273–1282, Fort Lauderdale, FL, April 2017.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *Proc. Intl. Conf. Machine Learn.*, pages 4615–4625, Long Beach, CA, June 2019.

Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

Chaoyue Niu, Fan Wu, Shaojie Tang, Lifeng Hua, Rongfei Jia, Chengfei Lv, Zhihua Wu, and Guihai Chen. Secure federated submodel learning. *arXiv preprint:1911.02254*, November 2019.

Zhimin Peng, Yangyang Xu, Ming Yan, and Wotao Yin. Arock: an algorithmic framework for asynchronous parallel coordinate updates. *SIAM J. Sci. Comp.*, 38(5):2851–2879, September 2016.

Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23 (2):1126–1153, June 2013.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Proc. Advances in Neural Info. Process. Syst.*, pages 693–701, Granada, Spain, December 2011.

C Okan Sakar, Gorkem Serbes, Aysegul Gunduz, Hunkar C Tunc, Hatice Nizam, Betul Erdogdu Sakar, Melih Tutuncu, Tarkan Aydin, M Erdem Isenkul, and Hulya Apaydin. A comparative analysis of speech signal processing algorithms for parkinson's disease classification and the use of the tunable q-factor wavelet transform. *Applied Soft Computing*, 74:255–263, 2019.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Proc. Conf. Intl. Speech Comm. Assoc.*, Singapore, Sept 2014.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Proc. Advances in Neural Info. Process. Syst.*, pages 4427–4437, Long Beach, CA, December 2017.

Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with memory. In *Proc. Advances in Neural Info. Process. Syst.*, pages 4447–4458, Montreal, Canada, Dec 2018.

Nikko Strom. Scalable distributed DNN training using commodity gpu cloud computing. In *Proc. Conf. Intl. Speech Comm. Assoc.*, Dresden, Germany, Sept 2015.

Chang Sun, Lianne Ippel, Johan van Soest, Birgit Wouters, Alexander Malic, Onaopepo Adekunle, Bob van den Berg, Ole Mussmann, Annemarie Koster, Carla van der Kallen, et al. A privacy-preserving infrastructure for analyzing personal health data in a vertically partitioned scenario. *Studies in health technology and informatics*, 264:373–377, 2019.

Tao Sun, Robert Hannah, and Wotao Yin. Asynchronous coordinate descent under more realistic assumptions. In *Proc. Advances in Neural Info. Process. Syst.*, pages 6183–6191, Long Beach, CA, December 2017.

Latanya Sweeney. k-anonymity: A model for protecting privacy. *Intl. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

Jianyu Wang and Gauri Joshi. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *arXiv preprint:1808.07576*, August 2018.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013.

Yangyang Xu and Wotao Yin. Block stochastic gradient iteration for convex and nonconvex optimization. *SIAM Journal on Optimization*, 25(3):1686–1716, 2015.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intelligent Systems and Technology*, 10(2), January 2019.

Andrew C Yao. Protocols for secure computations. In *Annual Symposium on Foundations of Computer Science*, pages 160–164, Chicago, Illinois, 1982.

Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proc. AAAI Conf. Artificial Intell.*, volume 33, pages 5693–5700, 2019.

Supplementary materials for “VAFL: a Method of Vertical Asynchronous Federated Learning"

Table of Contents

Appendix A. Supporting Lemmas

For notational brevity, we first define

$$G_m^k = \nabla_{\theta_m} F(\theta_0^k, \boldsymbol{\theta}^k) \tag{25a}$$

$$G_0^k = \nabla_{\theta_0} F(\theta_0^k, \boldsymbol{\theta}^k) \tag{25b}$$

$$\hat{g}_0^k = \nabla_{\theta_0} \ell(\theta_0, h_{n_k,1}^{k-\tau_{n_k,1}^k}, \ldots, h_{n_k,M}^{k-\tau_{n_k,M}^k}; y_{n_k}) \tag{25c}$$

$$\hat{g}_m^k = \nabla h_m(\theta_m^k; x_{n_k,m}) \nabla_{h_m} \ell(\theta_0^k, h_{n_k,1}^{k-\tau_{n_k,1}^k}, \ldots, h_{n_k,M}^{k-\tau_{n_k,M}^k}; y_{n_k}). \tag{25d}$$

We first establish the descent in the objective value.

Lemma 11 *Under Assumptions 1-3, the iterates $\{\theta_0^k, \boldsymbol{\theta}^k\}$ generated by Algorithm 1 satisfy*

$$\begin{aligned}\mathbb{E}[F^{k+1}|\Theta^k] \leq F^k &+ c_k \mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2|\Theta^k] + L(\eta_0^k)^2\sigma_0^2 \\ &+ \sum_{m=1}^{M} q_m L(\eta_m^k)^2 \sigma_m^2 - (\frac{\eta_0^k}{2} - L(\eta_0^k)^2)\|G_0^k\|^2 \\ &- \sum_{m=1}^{M} q_m (\frac{\eta_m^k}{2} - L(\eta_m^k)^2)\|G_m^k\|^2\end{aligned} \tag{26}$$

where $G_m^k := \nabla_{\theta_m} F(\theta_0^k, \boldsymbol{\theta}^k)$, $G_0^k := \nabla_{\theta_0} \ell(\theta_0^k, h_{n_k,1}^{k-\tau_{n_k,1}^k}, \ldots, h_{n_k,M}^{k-\tau_{n_k,M}^k}; y_{n_k})$, c_k is a constant depending on η_0^k, η_m^k with detailed expression specified in the supplementary materials, and Θ^k is the σ-algebra generated by $\{\theta_0^0, \boldsymbol{\theta}^0, \ldots, \theta_0^k, \boldsymbol{\theta}^k\}$.

Proof By Assumption 2, we have

$$\begin{aligned}
F^{k+1} =& F(\theta_0^k - \eta_0^k \hat{g}_0^k, \ldots, \theta_{m_k}^k - \hat{g}_{m_k}^k, \ldots) \\
\leq& F^k - \eta_0^k \langle G_0^k, \hat{g}_0^k \rangle\rangle - \eta_{m_k}^k \langle G_{m_k}^k, \hat{g}_{m_k}^k \rangle + \frac{L(\eta_0^k)^2}{2}\|\hat{g}_0^k\|^2 + \frac{L(\eta_{m_k}^k)^2}{2}\|\hat{g}_{m_k}^k\|^2 \\
\leq& F^k - \eta_0^k \langle G_0^k, g_0^k \rangle - \eta_0^k \langle G_0^k, \hat{g}_0^k - g_0^k \rangle - \eta_{m_k}^k \langle G_{m_k}^k, g_{m_k}^k \rangle - \eta_{m_k}^k \langle G_{m_k}^k, \hat{g}_{m_k}^k - g_{m_k}^k \rangle \\
&+ \frac{L(\eta_0^k)^2}{2}\|\hat{g}_0^k\|^2 + \frac{L(\eta_{m_k}^k)^2}{2}\|\hat{g}_{m_k}^k\|^2 \\
=& F^k - \eta_0^k \langle G_0^k, g_0^k \rangle - \eta_{m_k}^k \langle h_m^k, g_{m_k}^k \rangle + \frac{\eta_0^k}{2}\|G_0^k\|^2 + \frac{\eta_{m_k}^k}{2}\|G_{m_k}^k\|^2 + L(\eta_0^k)^2\|g_0^k\|^2 + L(\eta_{m_k}^k)^2\|g_{m_k}^k\|^2 \\
&+ (\frac{\eta_0^k}{2} + L(\eta_0^k)^2)\|\hat{g}_0^k - g_0^k\|^2 + (\frac{\eta_{m_k}^k}{2} + L(\eta_{m_k}^k)^2)\|\hat{g}_{m_k}^k - g_{m_k}^k\|^2
\end{aligned} \tag{27}$$

Note that

$$\begin{aligned}
\mathbb{E}\left[\|g_{m_k}^k\|^2 | \Theta^k\right] &= \mathbb{E}\left[\|g_{m_k}^k - G_{m_k}^k + G_{m_k}^k\|^2 | \Theta^k\right] \\
&= \mathbb{E}\left[\|g_{m_k}^k - G_{m_k}^k\|^2 | \Theta^k\right] + 2\mathbb{E}\left[\langle g_{m_k}^k - G_{m_k}^k, G_{m_k}^k \rangle | \Theta^k\right] + \|G_{m_k}^k\|^2 \\
&= \sigma_{m_k}^2 + \|G_{m_k}^k\|^2.
\end{aligned} \tag{28}$$

First we take expectation on (27) with respect to n_k, conditioned on Θ^k and $m_k = m$, we have

$$\begin{aligned}
\mathbb{E}[F^{k+1} | m_k = m, \Theta^k] \leq& F^k - (\frac{\eta_0^k}{2} - L(\eta_0^k)^2)\|G_0^k\|^2 - (\frac{\eta_m^k}{2} - L(\eta_m^k)^2)\|G_m^k\|^2 + L(\eta_0^k)^2\sigma_0^2 + L(\eta_m^k)^2\sigma_m^2 \\
&+ (\frac{\eta_0^k}{2} + L(\eta_0^k)^2)\mathbb{E}[\|\hat{g}_0^k - g_0^k\|^2 | m_k = m] + (\frac{\eta_m^k}{2} + L(\eta_m^k)^2)\mathbb{E}[\|\hat{g}_m^k - g_m^k\|^2 | m_k = m].
\end{aligned} \tag{29}$$

Then taking expectation with respect to m_k, it follows that

$$\begin{aligned}
\mathbb{E}[F^{k+1} | \Theta^k] \leq& F^k - (\frac{\eta_0^k}{2} - L(\eta_0^k)^2)\|G_0^k\|^2 - \sum_{m=1}^M q_m(\frac{\eta_m^k}{2} - L(\eta_m^k)^2)\|G_m^k\|^2 + L(\eta_0^k)^2\sigma_0^2 + \sum_{m=1}^M q_m L(\eta_m^k)^2\sigma_m^2 \\
&+ (\frac{\eta_0^k}{2} + L(\eta_0^k)^2)L_0^2\mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2 | \Theta^k] + \sum_{m=1}^M q_m(\frac{\eta_m^k}{2} + L(\eta_m^k)^2)L_m^2\mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2 | m_k = m, \Theta^k] \\
\leq& F^k - (\frac{\eta_0^k}{2} - L(\eta_0^k)^2)\|G_0^k\|^2 - \sum_{m=1}^M q_m(\frac{\eta_m^k}{2} - L(\eta_m^k)^2)\|G_m^k\|^2 + L(\eta_0^k)^2\sigma_0^2 + \sum_{m=1}^M q_m L(\eta_m^k)^2\sigma_m^2 \\
&+ \underbrace{\left(\left(\frac{\eta_0^k}{2} + L(\eta_0^k)^2\right)L_0^2 + \max_m\left(\frac{\eta_m^k}{2} + L(\eta_m^k)^2\right)L_m^2\right)}_{c^k}\mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2 | \Theta^k].
\end{aligned}$$

■

Appendix B. Convergence with bounded delay

When $\tau^k_{n_k,m} \leq D$, it can be derived that

$$\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2 \leq \sum_{d=1}^{D} D \left\|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\right\|^2. \tag{30}$$

B.1 Proof of Lemma 1

Recall the definition of V^k, that is

$$V^k = F^k + \sum_{d=1}^{D} \gamma_d \|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\|^2,$$

where we define $\boldsymbol{\theta}^{-D+1} = \cdots = \boldsymbol{\theta}^{-1} = \boldsymbol{\theta}^0$. To analyze the descent of V^k, we first decompose $\|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|^2$ as

$$\|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|^2 = (\eta^k_{m_k})^2 \|\hat{g}^k_{m_k}\|^2 \leq 2(\eta^k_{m_k})^2 \|g^k_{m_k}\|^2 + 2(\eta^k_{m_k})^2 \|\hat{g}^k_{m_k} - g^k_{m_k}\|^2. \tag{31}$$

Taking expectation on both sides of (31), and applying (28) leads to

$$\mathbb{E}\left[\|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|^2 | m_k = m, \Theta^k\right] \leq 2(\eta^k_m)^2 \|G^k_m\|^2 + 2(\eta^k_m)^2 \sigma^2_m + 2(\eta^k_m)^2 L^2_m \mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2 | m_k = m, \Theta^k] \tag{32}$$

$$\mathbb{E}\left[\|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|^2 | \Theta^k\right] \leq \sum_{m=1}^{M} 2q_m (\eta^k_m)^2 \|G^k_m\|^2 + \sum_{m=1}^{M} 2q_m (\eta^k_m)^2 \sigma^2_m + 2 \max_m (\eta^k_m)^2 L^2_m \mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2 | \Theta^k] \tag{33}$$

Following Lemma 11 and (33), V^k satisfies

$$\begin{aligned}
\mathbb{E}[V^{k+1}|\Theta^k] - V^k =& \mathbb{E}[F^{k+1}|\Theta^k] - F^k + \gamma_1 \mathbb{E}[\|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|^2 | \Theta^k] \\
&+ \sum_{d=1}^{D-1} (\gamma_{d+1} - \gamma_d) \|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\|^2 - \gamma_D \|\boldsymbol{\theta}^{k+1-D} - \boldsymbol{\theta}^{k-D}\|^2 \\
\leq& - \left(\frac{\eta^k_0}{2} - L(\eta^k_0)^2\right) \|G^k_0\|^2 - \sum_{m=1}^{M} q_m \left(\frac{\eta^k_m}{2} - L(\eta^k_m)^2 - 2\gamma_1 (\eta^k_m)^2\right) \|G^k_m\|^2 \\
&+ L(\eta^k_0)^2 \sigma^2_0 + \sum_{m=1}^{M} q_m (L + 2\gamma_1)(\eta^k_m)^2 \sigma^2_m \\
&+ \sum_{d=1}^{D-1} \left(Dc^k + D\gamma_1 \max_m 2(\eta^k_m)^2 L^2_m + \gamma_{d+1} - \gamma_d\right) \|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\|^2 \\
&+ \left(Dc^k + D\gamma_1 \max_m 2(\eta^k_m)^2 L^2_m - \gamma_D\right) \|\boldsymbol{\theta}^{k+1-D} - \boldsymbol{\theta}^{k-D}\|^2.
\end{aligned} \tag{34}$$

Since we choose $\eta_0^k, \eta_m^k \leq \bar{\eta} \leq \frac{1}{4(L+2\gamma_1)}$, it follows that $c^k \leq \frac{3}{2}\bar{\eta}L^2$. By taking expectation on both sides of (34), we have

$$\begin{aligned}\mathbb{E}V^{k+1} - \mathbb{E}V^k \leq & -\frac{1}{4}\min\{\eta_0^k, q_m\eta_m^k\}\mathbb{E}[\|G^k\|^2] + L(\eta_0^k)^2\sigma_0^2 + (2\gamma_1 + L)\sum_{m=1}^M q_m(\eta_m^k)^2\sigma_m^2 \\ & - \sum_{d=1}^{D-1}(\gamma_d - \gamma_{d+1} - \frac{3}{2}D\bar{\eta}L^2 - 2D\gamma_1\bar{\eta}^2L^2)\mathbb{E}\|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\|^2 \\ & - (\gamma_D - \frac{3}{2}D\bar{\eta}L^2 - 2D\gamma_1\bar{\eta}^2L^2)\mathbb{E}\|\boldsymbol{\theta}^{k+1-D} - \boldsymbol{\theta}^{k-D}\|^2. \end{aligned} \tag{35}$$

B.2 Proof of Theorem 2

Let $\gamma_1 = \frac{\frac{3}{2}\bar{\eta}D^2L^2}{1-2D^2\bar{\eta}^2L^2} \leq \frac{1}{2}DL$,$\eta_0^k = \eta_m^k = \eta = \min\{\frac{1}{4(D+1)L}, \frac{c_\eta}{\sqrt{K}}\}$. Select $\gamma_2, \ldots, \gamma_D$ such that

$$\begin{aligned}&\gamma_d - \gamma_{d+1} - \frac{3}{2}D\eta L^2 - 2D\gamma_1\eta^2L^2 = 0, \quad d = 1, \ldots, D-1, \\ &\gamma_D - \frac{3}{2}D\eta L^2 - 2D\gamma_1\eta^2L^2 = 0.\end{aligned}$$

It can be verified that $\gamma_d \geq 0$. Then (35) reduces to

$$\mathbb{E}V^{k+1} - \mathbb{E}V^k \leq -\frac{1}{4}\min_m q_m\eta\mathbb{E}[\|G^k\|^2] + \eta^2L\sigma_0^2 + \eta^2(2\gamma_1 + L)\sum_{m=1}^M q_m\sigma_m^2. \tag{36}$$

By taking sum $\sum\limits_{k=0}^{K-1}\mathbb{E}V^{k+1} - \mathbb{E}V^k$, it follows that

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|G^k\|^2] \leq & \frac{F^0 - F^* + K\eta^2L\sigma_0^2 + K\eta^2(2\gamma_1 + L)\sum\limits_{m=1}^M q_m\sigma_m^2}{\frac{1}{4}\min\limits_m q_m\eta K} \\ \leq & \frac{16DL(F^0 - F^*)}{\min\limits_m q_m K} + \frac{4c_\eta(F^0 - F^*)}{\min\limits_m q_m\sqrt{K}} + \frac{4c_\eta L\sigma_0^2 + c_\eta(8\gamma_1 + 4L)\sum\limits_{m=1}^M q_m\sigma_m^2}{\min\limits_m q_m\sqrt{K}}.\end{aligned}$$

B.3 Proof of Theorem 3

By strong convexity of $F(\theta_0, \boldsymbol{\theta})$, we have

$$2\mu(F(\theta_0, \boldsymbol{\theta}) - F^*) \leq \|\nabla F(\theta_0, \boldsymbol{\theta})\|^2.$$

Choose γ_d such that

$$\begin{aligned}&\gamma_d - \gamma_{d+1} - \frac{3}{2}D\bar{\eta}L^2 - 2D\gamma_1\bar{\eta}^2L^2 = \frac{\mu}{2}\min_m q_m\bar{\eta}\gamma_1, \quad d = 1, \ldots, D-1, \\ &\gamma_D - \frac{3}{2}D\bar{\eta}L^2 - 2D\gamma_1\bar{\eta}L^2 = \frac{\mu}{2}\min_m q_m\bar{\eta}\gamma_1.\end{aligned}$$

Solve the linear equations above and get

$$\gamma_1 = \frac{\frac{3}{2}\bar{\eta}D^2L^2}{1 - 2D^2\bar{\eta}^2L^2 - \frac{\mu}{2}\min\limits_m q_m D\bar{\eta}}.$$

If $\bar{\eta} \leq \frac{1}{4(D+1)L+2\mu \min\limits_m q_m D}$, $\gamma_1 \leq 2\bar{\eta}D^2L^2$. Then (35) reduces to

$$\mathbb{E}V^{k+1} \leq (1-\frac{\mu}{2}\eta^k \min_m q_m)\mathbb{E}V^k + (\eta^k)^2 \underbrace{\left(L\sigma_0^2 + (2\gamma_1+L)\sum_{m=1}^M q_m\sigma_m^2\right)}_{R}.$$

Let $\eta^k = \frac{4}{\mu \min\limits_m q_m(k+K_0)}$ where $K_0 = \frac{4(4(D+1)L+2\mu \min\limits_m q_m D)}{\mu \min\limits_m q_m}$. Then

$$\begin{aligned}
\mathbb{E}V^k &\leq \prod_{k=0}^{K-1}(1-\frac{\mu}{2}\min_m q_m\eta^k)V^0 + R\sum_{k=0}^{K-1}(\eta^k)^2\prod_{j=k+1}^{K-1}(1-\frac{\mu}{2}\min_m q_m\eta_j)\\
&\leq \frac{(K_0-2)(K_0-1)}{(K+K_0-2)(K+K_0-1)}V^0 + \frac{16R}{\mu^2\min\limits_m q_m}\sum_{k=0}^{K-1}\frac{1}{(k+K_0)^2}\frac{(k+K_0-1)(k+K_0)}{(K+K_0-2)(K+K_0-1)}\\
&\leq \frac{(K_0-1)^2}{(K+K_0-1)^2}(F^0-F^*) + \frac{16RK}{\mu^2\min\limits_m q_m(K+K_0-1)^2}.
\end{aligned}$$

Appendix C. Convergence with stochastic unbounded delay

Before proceeding to the main proofs, we first present a useful fact. Given the definition of $p_{m,d}$ in Assumption 5, it can be verified that

$$\begin{aligned}
c_{m,d} &:= \sum_{s=d}^{\infty} s p_{m,s} = \bar{p}_m\left(\frac{d\rho^d}{1-\rho} + \frac{\rho^{d+1}}{(1-\rho)^2}\right),\\
\sum_{s=d}^{\infty} c_{m,s} &= \bar{p}_m\left(\frac{d\rho^d}{(1-\rho)^2} + \frac{2\rho^{d+1}}{(1-\rho)^3}\right),\\
c_m &:= \sum_{d=1}^{\infty} c_{m,d} = \bar{p}_m\left(\frac{\rho}{(1-\rho)^2} + \frac{2\rho^2}{(1-\rho)^3}\right).
\end{aligned}$$

For unbounded delay, we have the following relation

$$\begin{aligned}
\mathbb{E}[\|\hat{\boldsymbol{\theta}}^k - \boldsymbol{\theta}^k\|^2|\Theta^k] &\leq \sum_{m=1}^M\sum_{s=1}^{+\infty}\sum_{d=1}^{s} s p_{m,s}\|\theta_m^{k+1-d} - \theta_m^{k-d}\|^2\\
&= \sum_{m=1}^M\sum_{d=1}^{+\infty} c_{m,d}\|\theta_m^{k+1-d} - \theta_m^{k-d}\|^2
\end{aligned}$$

where the constant $c_{m,d}$ is defined as $c_{m,d} := \sum\limits_{s=d}^{+\infty} s p_{m,s}$.

C.1 Proof of Lemma 12

Lemma 12 *Under Assumptions 1, 2, 3 and 5, if we choose* $\eta_0^k \leq \frac{1}{4L}, \eta_m^k \leq \frac{1}{4(L+2\gamma_{m,1})}$*, then we have*

$$\begin{aligned}\mathbb{E}V^{k+1} - \mathbb{E}V^k \leq & -\left(\frac{\eta_0^k}{2} - L(\eta_0^k)^2\right)\|G_0^k\|^2 \\ & - \sum_{m=1}^M q_m \left(\frac{\eta_m^k}{2} - (2\gamma_{m,1}+L)(\eta_m^k)^2\right)\|\nabla_{\theta_m} F(\theta_0^k, \boldsymbol{\theta}^k)\|^2 \\ & + \sum_{m=1}^M \sum_{d=1}^k \left(\left(c^k + 2q_m\gamma_{m,1}(\eta_m^k)^2 L^2\right) c_{m,d} + \gamma_{m,d+1} - \gamma_{m,d}\right) \qquad (37) \\ & \times \mathbb{E}[\|\theta_m^{k+1-d} - \theta_m^{k-d}\|^2] \qquad (38) \\ & + L(\eta_0^k)^2\sigma_0^2 + \sum_{m=1}^M q_m(\eta_m^k)^2(2\gamma_{m,1}+L)\sigma_m^2. \qquad (39)\end{aligned}$$

Similar to (33), we can decompose the difference term as

$$\mathbb{E}\left[\|\theta_m^{k+1} - \theta_m^k\|^2 | \Theta^k\right] \leq 2q_m(\eta_m^k)^2\|G_m^k\|^2 + 2q_m(\eta_m^k)^2\sigma_m^2 + 2q_m(\eta_m^k)^2L^2\mathbb{E}[\|\hat{\theta}_m^k - \theta_m^k\|^2|\Theta^k]. \qquad (40)$$

Following Lemma 11 and (40), we have

$$\begin{aligned}&\mathbb{E}[V^{k+1}|\Theta^k] - V^k \\ =&\mathbb{E}[F^{k+1}|\Theta^k] - F^k + \sum_{m=1}^M \gamma_{m,1}\mathbb{E}[\|\theta_m^{k+1} - \theta_m^k\|^2|\Theta^k] + \sum_{m=1}^M\sum_{d=1}^\infty (\gamma_{d+1} - \gamma_d)\|\theta_m^{k+1-d} - \theta_m^{k-d}\|^2 \\ \leq & -\left(\frac{\eta_0^k}{2} - L(\eta_0^k)^2\right)\|G_0^k\|^2 - \sum_{m=1}^M q_m\left(\frac{\eta_m^k}{2} - (2\gamma_{m,1}+L)(\eta_m^k)^2\right)\|G_m^k\|^2 \\ & + \sum_{m=1}^M\sum_{d=1}^k \left(\left(c^k + 2q_m\gamma_{m,1}(\eta_m^k)^2L^2\right)c_{m,d} + \gamma_{m,d+1} - \gamma_{m,d}\right)\|\theta_m^{k+1-d} - \theta_m^{k-d}\|^2 \\ & + L(\eta_0^k)^2\sigma_0^2 + \sum_{m=1}^M q_m(\eta_m^k)^2(2\gamma_{m,1}+L)\sigma_m^2.\end{aligned}$$

Let $\eta_0^k, \eta_m^k \leq \bar{\eta} \leq \frac{1}{4(L+2\max_m \gamma_{m,1})}$. Then $c^k \leq \frac{3}{2}\bar{\eta}L^2$. By direct calculation, we have

$$\begin{aligned}\mathbb{E}V^{k+1} - \mathbb{E}V^k \leq & -\frac{1}{4}\min\{\eta_0^k, q_m\eta_m^k\}\mathbb{E}[\|G^k\|^2] + L(\eta_0^k)^2\sigma_0^2 + \sum_{m=1}^M q_m(\eta_m^k)^2(2\gamma_{m,1}+L)\sigma_m^2 \\ & - \sum_{m=1}^M\sum_{d=1}^\infty\left(\gamma_{m,d} - \gamma_{m,d+1} - c_{m,d}(\frac{3}{2}\bar{\eta}L^2 + 2q_m\gamma_{m,1}\bar{\eta}^2L^2)\right)\mathbb{E}[\|\theta_m^{k+1-d} - \theta_m^{k-d}\|^2].\end{aligned}$$

Select $\gamma_{m,d}$ such that

$$\left(\frac{3}{2}\bar{\eta}L^2 + 2q_m\gamma_{m,1}\bar{\eta}^2L^2\right)c_{m,d} + \gamma_{m,d+1} - \gamma_{m,d} = -\delta_m c_{m,d}, \quad m = 1,\ldots,M,\ d = 1,\ldots,\infty.$$

Then it remains that

$$\gamma_{m,d} = \sum_{s=d}^\infty c_{m,s}\left(\frac{3}{2}\bar{\eta}L^2 + 2q_m\gamma_{m,1}\bar{\eta}^2L^2 + \delta_m\right).$$

C.2 Proof of Theorem 4

We set $\delta_m = 0$ and

$$c_m = \sum_{d=1}^{\infty} c_{m,d}, \gamma_{m,1} = \frac{\frac{3}{2}c_m\bar{\eta}L^2}{1-2c_m q_m\bar{\eta}^2L^2} \le 2\bar{\eta}c_m L^2 \le \frac{1}{2}\sqrt{c_m}L \tag{41}$$

and

$$\eta_0^k = \eta_m^k = \eta = \min\left\{\frac{1}{4(1+\max_m\sqrt{c_m})L}, \frac{c_\eta}{\sqrt{K}}\right\}. \tag{42}$$

Plugging these constants, we have

$$\mathbb{E}V^{k+1} - \mathbb{E}V^k \le \frac{1}{4}\min_m q_m\eta\mathbb{E}[\|G^k\|^2] + \eta^2\sum_{m=1}^M q_m(2\gamma_{m,1}+L)\sigma_m^2.$$

By taking sum $\sum\limits_{k=0}^{K-1}\mathbb{E}V^{k+1} - \mathbb{E}V^k$, it follows that

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|G^k\|^2] &\le \frac{F^0 - F^* + K\eta^2\sum\limits_{m=1}^M(2\gamma_{m,1}+L)q_m\sigma_m^2}{\frac{1}{4}\min\limits_m q_m\eta K} \\ &\le \frac{16cL(F^0-F^*)}{\min\limits_m q_m K} + \frac{4c_\eta(F^0-F^*)}{\min\limits_m q_m\sqrt{K}} + \frac{4c_\eta\sum\limits_{m=1}^M(2\gamma_{m,1}+L)q_m\sigma_m^2}{\min\limits_m q_m\sqrt{K}}.\end{aligned}$$

C.3 Proof of Theorem 5

First derive $\gamma_{m,1} = \frac{\frac{3}{2}c_m\bar{\eta}L^2+\delta}{1-2c_m\bar{\eta}^2L^2} = 2c_m\bar{\eta}L^2 \le \frac{1}{2}\sqrt{c_m}L$ by setting $\delta_m = \frac{1}{4}c_m\bar{\eta}L^2$. Let $\nu = \min\{\frac{\delta_m c_{m,d}}{\bar{\eta}\gamma_{m,d}}, \frac{\mu q_m}{2}\}$, $\eta^k \le \bar{\eta} = \frac{1}{4(1+\max_m\sqrt{c_m})L}$ and get

$$\mathbb{E}V^{k+1} \le (1-\nu\eta^k)\mathbb{E}V^k + (\eta^k)^2\underbrace{\left(L\sigma_0^2 + \sum_{m=1}^M(2\gamma_{m,1}+L)q_m\sigma_m^2\right)}_{:=R}.$$

To determine ν, note that

$$\begin{aligned}\frac{\delta_m c_{m,d}}{\bar{\eta}\gamma_{m,d}} &= \frac{c_{m,d}}{\sum\limits_{s=d}^{\infty}c_{m,s}}\frac{1}{\bar{\eta}}\frac{\delta_m}{\frac{3}{2}\bar{\eta}L^2 + 2q_m\gamma_{m,1}\bar{\eta}^2L^2 + \delta_m} \ge \frac{1}{\bar{\eta}}\frac{(1-\rho)\delta_m}{3\bar{\eta}L^2 + 4q_m\gamma_{m,1}\bar{\eta}^2L^2 + 2\delta_m} \\ &\ge \frac{1}{\bar{\eta}}\frac{(1-\rho)\delta_m}{5\bar{\eta}L^2 + \delta_m} = \frac{1}{\bar{\eta}}\frac{(1-\rho)c_m}{20+c_m}\end{aligned}$$

where we use the fact that $\frac{c_{m,d}}{\sum\limits_{s=d}^{\infty} c_{m,s}} \geq \frac{1-\rho}{2}$. Let $\eta^k = \frac{2}{\nu(k+K_0)}$ where $K_0 = \frac{4(1+\max_m \sqrt{c_m})L}{\nu}$. Then

$$
\begin{aligned}
\mathbb{E}V^k &\leq \prod_{k=0}^{K-1}(1-\nu\eta^k)V^0 + R\sum_{k=0}^{K-1}(\eta^k)^2 \prod_{j=k+1}^{K-1}(1-\nu\eta_j)\\
&\leq \frac{(K_0-2)(K_0-1)}{(K+K_0-2)(K+K_0-1)}V^0 + \frac{16R}{\mu^2 \min\limits_m q_m}\sum_{k=0}^{K-1}\frac{1}{(k+K_0)^2}\frac{(k+K_0-1)(k+K_0)}{(K+K_0-2)(K+K_0-1)}\\
&\leq \frac{(K_0-1)^2}{(K+K_0-1)^2}(F^0-F^*) + \frac{16RK}{\mu^2 \min\limits_m q_m (K+K_0-1)^2}.
\end{aligned}
$$

Appendix D. Proof of Theorem 6

Before proceeding to the proof of Theorem 6, we first present the smoothness of a single neuron in the following lemma.

Lemma 13 *If $\sigma(x)$ is L_σ^0-Lipschitz continuous and differentiable almost everywhere, Z is a continuous random variable with pdf $\mu(z)$, then $\bar{\sigma}(x) := \mathbb{E}\sigma(x+Z)$ is differentiable with Lipschitz continuous gradient $\nabla\bar{\sigma}(x) = \mathbb{E}\nabla\sigma(x+Z)$.*

Proof We first prove that $\bar{\sigma}(x)$ is smooth and $\mathbb{E}\nabla\sigma(x+Z) = \nabla\bar{\sigma}(x)$.

$$
\frac{\mathbb{E}_Z\sigma(x+\delta v+Z) - \mathbb{E}_Z\sigma(x+Z)}{\delta} = \int_{\mathbb{R}^d}\frac{\sigma(x+\delta v+z)-\sigma(x+z)}{\delta}\mu(z)dz
$$

Since σ is differentiable almost everywhere, for any fixed $x \in \mathbb{R}^d$ and directional vector $v \in \mathbb{R}^d$,

$$
\lim_{\delta\to 0}\frac{\sigma(x+\delta v+z)-\sigma(x+z)}{\delta} = v^\top\nabla\sigma(x+z) = \sum_{i=1}^{n}\frac{\partial\sigma}{\partial x_i}(x+z)v_i \qquad a.e.
$$

and

$$
\int_{\mathbb{R}^d}\left|\frac{\sigma(x+\delta v+z)-\sigma(x+z)}{\delta}\right|\mu(z)dz \leq \int_{\mathbb{R}^d} L_\sigma^0\mu(z)dz = L_\sigma^0.
$$

Then by dominated convergence theorem, when taking $\delta \to 0$,

$$
\begin{aligned}
\frac{\partial\bar{\sigma}}{\partial x_i}(x) &= \int_{\mathbb{R}^d}\frac{\partial\sigma}{\partial x_i}(x+z)\mu(z)dz,\\
\frac{\partial\bar{\sigma}}{\partial v}(x) &= \int_{\mathbb{R}^d}\sum_{i=1}^{d}\frac{\partial\sigma}{\partial x_i}(x+z)v_i\mu(z)dz = \sum_{i=1}^{n}\frac{\partial\bar{\sigma}}{\partial x_i}(x)v_i.
\end{aligned}
$$

Therefore, $\bar{\sigma}(x)$ is differentiable,

$$
\nabla\bar{\sigma}(x) = \int_{\mathbb{R}^d}\nabla\sigma(x+z)\mu(z)dz = \mathbb{E}_Z\nabla\sigma(x+Z).
$$

Next we derive the smoothness constant of $\bar{\sigma}(x)$. We focus on the uniform distribution and Gaussian distribution.

I. Assume that the uniform distribution $Z \sim \mathcal{U}[-\frac{c}{2}, \frac{c}{2}]^d$, i.e., $\mu(z) = \frac{1}{c^d}\mathbb{1}_{\{-\frac{c}{2} \le z_i \le \frac{c}{2}, 1 \le i \le d\}}(z)$.

$$\begin{aligned}
\|\nabla\bar{\sigma}(x) - \nabla\bar{\sigma}(x')\| &= \left\| \int_{\mathbb{R}^d} \nabla\sigma(x+y)\mu(z)dz - \int_{\mathbb{R}^d} \nabla\sigma(x'+z)\mu(z)dz \right\| \\
&= \left\| \int_{\mathbb{R}^d} \nabla\sigma(y)(\mu(y-x) - \mu(y-x'))dy \right\| \le L_\sigma^0 \int_{\mathbb{R}^d} |\mu(y-x) - \mu(y-x')|\, dy \\
&\le \frac{2\sqrt{d}L_\sigma^0}{c}\|x - x'\| := L_{\bar{\sigma}}(c)\|x - x'\|
\end{aligned}$$

where the smoothness constant is defined as $L_{\bar{\sigma}}(c) := \frac{2\sqrt{d}L_\sigma^0}{c}$.

II. Assume that the Gaussian distribution $z \sim \mathcal{N}(0, c^2 I_d)$, i.e., $\mu(z) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{\|z\|^2}{2\sigma^2}}$.

$$\bar{\sigma}(x) = \int_{\mathbb{R}^d} \sigma(x+z)\mu(z)dz = \int_{z\in\mathbb{R}^d} \sigma(y)\mu(y-x)dy.$$

By Leibniz rule, we have

$$\nabla\bar{\sigma}(x) = -\int_{\mathbb{R}^d} \sigma(y)\nabla\mu(y-x)dy = -\int_{\mathbb{R}^d} \sigma(y+x)\nabla\mu(y)dy.$$

Then it follows

$$\begin{aligned}
\|\nabla\bar{\sigma}(x) - \nabla\bar{\sigma}(x')\| &= \left\| \int_{\mathbb{R}^d} (\sigma(y+x) - \sigma(y+x'))\nabla\mu(y)dy \right\| \\
&\le L_\sigma^0 \left(\int_{\mathbb{R}^d} \|\nabla\mu(y)\|dy \right) \|x - x'\| \\
&= \frac{L_\sigma^0 d}{c}\|x - x'\| := L_{\bar{\sigma}}(c)\|x - x'\|
\end{aligned}$$

where the smoothness constant is $L_{\bar{\sigma}}(c) := \frac{L_\sigma^0 d}{c}$. ■

D.1 Proof of Theorem 6

For simplicity, we assume that all the activation functions are same, e.g., $\sigma_l = \sigma, \forall l = 1, \cdots, L$. We use L_f to denote the lipschitz constant of a scalar or vector function f. In the following proof, we change the order of differentiation and integration (expectation) as it is supported by Leibniz integral rule. We also let $\bar{f} = \mathbb{E}f$.

Since $\nabla_{b_L}\bar{h} = \mathbb{E}[\nabla\bar{\sigma}], \nabla_{w_L}\bar{h} = \mathbb{E}[\nabla\bar{\sigma}]u_{L-1}^\top, \nabla_{u_{L-1}}\mathbb{E}_{Z_L}h = w_L^\top\mathbb{E}[\nabla\bar{\sigma}]$. The smoothness of $\bar{\sigma}$ implies that $\nabla_{b_L}\bar{h}, \nabla_{w_L}\bar{h}, \nabla_{u_{L-1}}\bar{h}$ are $L_{b_L}^{\bar{h}}, L_{w_L}^{\bar{h}}, L_{u_{L-1}}^{\bar{h}}$-Lipschitz continuous respectively, with

$$\begin{aligned}
&L_{b_L}^{\bar{h}} = L_{\bar{\sigma}}(c), \\
&L_{w_L}^{\bar{h}} = L_{b_L}^{\bar{h}}\mathbb{E}[\|u_{L-1}\|], \\
&L_{u_{L-1}}^{\bar{h}} = L_{b_L}^{\bar{h}}\|w_L\|, \\
&\|\nabla_{u_{L-1}}\bar{h}\| \le L_\sigma^0\|w_L\|.
\end{aligned}$$

Since σ is differentiable almost everywhere, $\bar{\sigma}(w_L\sigma(\cdot + Z_{L-1}) + b_L)$ is differentiable almost everywhere and thus is smooth in expectation of Z_{L-1}. By some calculation, we can show that

$$\begin{aligned}
L^{\bar{h}}_{b_{L-1}} &= L^{\bar{h}}_{u_{L-1}}(L^0_\sigma)^2 + L^0_\sigma\|w_L\|L_{\bar{\sigma}}(c)\\
L^{\bar{h}}_{w_{L-1}} &= L^{\bar{h}}_{b_{L-1}}\mathbb{E}[\|u_{L-1}\|]\\
L^{\bar{h}}_{u_{L-2}} &= L^{\bar{h}}_{b_{L-1}}\|w_{L-1}\|
\end{aligned}$$

Following the similar steps, we obtain that

$$\begin{aligned}
L^{\bar{h}}_{b_l} &= L^{\bar{h}}_{u_l}(L^0_\sigma)^2 + L^0_\sigma\|w_L\|\cdots L^0_\sigma\|w_{l+1}\|L_{\bar{\sigma}}(c),\\
L^{\bar{h}}_{w_l} &= L^{\bar{h}}_{b_l}\mathbb{E}[\|u_{l-1}\|],\\
L^{\bar{h}}_{u_{l-1}} &= L^{\bar{h}}_{b_l}\|w_l\|,
\end{aligned}$$

As long as the overall loss function $\ell(\theta_0, h_1, \ldots, h_M; y)$ is smooth w.r.t. $\theta_0, h_1, \ldots, h_M$, we can extend our results to show that it is smooth in the local parameters $\theta_1, \ldots, \theta_M$. Take u_l from h_m as example,

$$L^{\bar{\ell}}_{\theta_m} = L^{\bar{\ell}}_{h_m}(L^0_{h_m})^2 + L^0_\ell L^{\bar{h}_m}_{\theta_m}$$

we can extend our results to show that $F(\theta_0, \boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\ell(\theta_0, h_{n,1}, \ldots, h_{n,M}; y_n) + \sum_{m=1}^{M} r(\theta_m)$ is smooth, where the expectation is taken with respect to all the random neurons in local embeddings $h_1, \ldots, h_M$.

D.2 The objective difference after local perturbation

Now we evaluate the difference between $F_{\nu,c}(\theta_0, \boldsymbol{\theta})$ and $F(\theta_0, \boldsymbol{\theta})$. Note that

$$\begin{aligned}
|F_{\nu,c}(\theta_0, \boldsymbol{\theta}) - F(\theta_0, \boldsymbol{\theta})|^2 &= \left|\frac{1}{N}\sum_{n=1}^{N}(\mathbb{E}_{\mathbf{Z}}\ell(\theta_0, h'_{n,1}, \ldots, h'_{n,M}; \mathbf{Z}) - \ell(\theta_0, h_{n,1}, \ldots, h_{n,M}))\right|^2\\
&\le \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{\mathbf{Z}}[|\ell(\theta_0, h'_{n,1}, \ldots, h'_{n,M}; \mathbf{Z}) - \ell(\theta_0, h_{n,1}, \ldots, h_{n,M})|^2]\\
&\le \frac{M}{N}\sum_{n=1}^{N}L^2_\ell\mathbb{E}_{\mathbf{Z}}[\|h'_{n,m} - h_{n,m}\|^2] \qquad (43)
\end{aligned}$$

where $h_{n,m}$ and $h'_{n,m}$ correspond to the outputs of (18) and (19), respectively. Then it follows that

$$\|h'_{n,m} - h_{n,m}\| = \|u'_L - u_L\| \le L_{\sigma_L}(\|w_L\|\|u'_{L-1} - u_{L-1}\| + \|Z_L\|) \le \cdots \le \sum_{j=1}^{L}\left(\prod_{l=j}^{L}L_{\sigma_l}\|w_l\|\right)\|Z_l\|$$

$$\|h'_{n,m} - h_{n,m}\|^2 \le \left(\sum_{j=1}^{L}\prod_{l=j}^{L}L^2_{\sigma_l}\|w_l\|^2\right)\left(\sum_{j=1}^{L}\|Z_j\|^2\right).$$

$$\mathbb{E}[\|h'_{n,m} - h_{n,m}\|^2] \le \left(\sum_{j=1}^{L}\prod_{l=j}^{L}L^2_{\sigma_l}\|w_l\|^2\right)\left(\sum_{j=1}^{L-1}\frac{c_l^2}{12} + \nu^2\right)$$

Plugging into (43), we arrive at

$$|F_{\nu,c}(\theta_0, \boldsymbol{\theta}) - F(\theta_0, \boldsymbol{\theta})| \le M \left(\sum_{j=1}^{L} \prod_{l=j}^{L} L_{\sigma_l}^2 \|w_l\|^2 \right)^{\frac{1}{2}} \left(\sum_{j=1}^{L-1} \frac{c_l^2}{12} + \nu^2 \right)^{\frac{1}{2}} .$$

Appendix E. Proof of Theorem 9

Let u_l, u_l' denote the the outputs of l-th layer with inputs $u_0 = x, x'$. Under the assumptions that $Z_l \sim \mathcal{U}[-c_l/2, c_l/2]$,$\sigma_l$ is L_{σ_l}-Lipschitz continuous for $l = 1, \dots, L-1$, we can derive that

$$\begin{aligned} \|w_L u_{L-1} - w_L u_{L-1}'\| &\le \|w_L\| L_{\sigma_{L-1}} (\|w_{L-1}\| \|u_{L-2} - u_{L-2}'\| + \sqrt{d_{L-1}} c_{L-1}) \\ &\le \|w_L\| \prod_{l=1}^{L-1} L_{\sigma_l} \|w_l\| \|x - x'\| + \|w_L\| \sum_{l=1}^{L-1} \left(\prod_{j=1}^{l} L_{\sigma_j} \sqrt{d_j} \right) c_l \\ &:= \bar{B}. \end{aligned}$$

Consider the linear operation of L-th layer $\mathcal{M}(u_{L-1}) = w_L u_{L-1} + b_L + Z_L$ which is a random mechanism defined by $Z_L \sim \mathcal{N}(0, \nu^2)$. Since differential privacy is immune to post-processing (Dwork et al., 2014), $\sigma_L \circ \mathcal{M}$ does not increase the privacy loss compared with $\mathcal{M}$. According to Theorem 1 in (Abadi et al., 2016), Algorithm 1 is (ε, δ)-differentially private if $\nu = c\frac{q\sqrt{T\log(1/\delta)}}{\varepsilon}$.

Appendix F. Additional simulations

F.1 Asynchronous VFL via logistic regression

The data are distributed across $M = 8$ clients for CIFAR-10 and $M = 7$ for MNIST. For each client, the batch size is selected to be 0.01 of the whole amount of training dataset. We choose stepsize $\eta = 1 \times 10^{-4}$ for both CIFAR-10 and MNIST. In the simulation, we add a random delay following the Poisson distribution representing the difference of computing capability between different local client.

For CIFAR-10 and MNIST, we both train the training dataset sufficient epochs and record the loss, accuracy and wall clock time at the end of each epoch. Numerical results are reported in Figures and . From both Figures and , we find that the loss of asynchronous training converges faster than the synchronous one always reaches the same loss value in less time.

we found that in the same time period, local clients applying asynchronous method update local parameter more times than the synchronous one. Comparing and , CIFAR-10 tends to have a training curve gap between two algorithm than MNIST because it has a flatter loss curve than MNIST. We can infer that asynchronous method has a larger advantage than synchronous one when the dataset is not easy to be trained with the current model or there exists a significant computing capability or communication efficiency difference between clients.

F.2 Asynchronous VFL via nonlinear local embedding

We train a parameter server structured convolutional neural network which consists of two parts: the local embedding part and the server part. The former one is composed of 12 clients each having a 7-layer convolutional neural network shown in . The server part is a centralized two layer fully connect neural network shown in . The data we choose is Modelnet40 and we vertically distributed images of the objects in the dataset from 12 angles and assign to each local client. Each local client deals with the data assigned by their local convolutional network and generate a vector whose dimension is 512 as the local output.

The server then combine the 12 vectors linearly and send them into the two-layer fully connected neural network and classify into 40 classes.

We choose stepsize as $\eta = 10^{-3}$ and we get the following numerical test result in Figure 5 and 6. We also add a random delay on each client in both the asynchronous and synchronous method. The advantage of asynchronous algorithm is obvious. It takes less time to reach a lower loss. The each of the client optimize their local parameters independently. Computation and optimization efficiency in a single unit time has been increased significantly. The training accuracy of asynchronous algorithm almost reaches 100% in the late training process, which infers that asynchronous trends to converge quicker and performs better.

MIMIC is an open dataset developed by the MIT Lab for Computational Physiology, comprising deidentified health data associated with 60,000 intensive care unit admissions.