Distributed Minimax Fair Optimization over Hierarchical Networks

Wen Xu*, Juncheng Wang†, Ben Liang*, Gary Boudreau‡, Hamza Sokun‡
*University of Toronto, Canada, †Hong Kong Baptist University, Hong Kong, ‡Ericsson, Canada

ABSTRACT

In modern applications, the underlying computation and communication networks are often hierarchical, which is typified by the three-layer client-edge-cloud system that has become prominent in recent times. We study minimax fairness in distributed optimization over such systems, to provide robust performance guarantee for the worst-case mixture of loss functions. We propose HierMinimax, a communication efficient distributed algorithm to solve the minimax optimization problem. We provide convergence analysis for both convex and non-convex loss functions, leading to performance bounds that enable tuning the tradeoff between the communication complexity and the optimization convergence rate. Our experiments on classification problems with canonical datasets show that HierMinimax substantially improves the fairness in learning accuracy and reduces the communication overhead compared with the current best alternatives.

CCS CONCEPTS

• **Computing methodologies** → **Distributed algorithms**; **Machine learning**; • **Networks**;

KEYWORDS

Minimax optimization, distributed learning, hierarchical networks

ACM Reference Format:
Wen Xu, Juncheng Wang, Ben Liang, Gary Boudreau, and Hamza Sokun. 2024. Distributed Minimax Fair Optimization over Hierarchical Networks. In *The 53rd International Conference on Parallel Processing (ICPP '24), August 12–15, 2024, Gotland, Sweden.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3673038.3673137

1 INTRODUCTION

Our work is motivated by federated learning (FL) [23] as a prime example of distributed optimization. In FL, multiple local clients collaboratively train a machine learning (ML) model with the assistance of a server. The standard objective in FL is to find a best global model $\boldsymbol{w}$ that minimizes a weighted sum of the local loss functions at individual clients:

$$\min_{\boldsymbol{w}} \sum_{n\in\mathcal{N}} q_n f_n(\boldsymbol{w}), \tag{1}$$

where $\mathcal{N}$ is the set of clients, $f_n(\boldsymbol{w})$ is the local loss function at the nth client, and $q_n \geq 0$ is some given local weight. It is common to choose q_n such that it is proportional to the amount of data at the nth client.

FL is typically considered in a *two-layer client-server* architecture [14, 18, 38]. In edge computing, the clients are mobile or IoT devices and the server can be either an edge server or a cloud server [3]. Exemplary algorithms under this architecture include Federated Averaging (FedAvg) [23] and Local Stochastic Gradient Descent (Local-SGD) [34]. For each training round, each client updates its local model via SGD from a common global model using its local data, and then the server aggregates all the updated local models to generate a new global model.

However, in many practical applications, the underlying communication networks are multi-layer hierarchical. An example is to perform FL as a cloud-based distributed service, where the clients are small remote devices and the server is in some far away cloud center [3]. Therefore, the communication between the clients and the server must go through some intermediate edge server. Such hierarchical communication architecture was ignored in [3, 14, 18, 23, 34, 38]. In contrast, a *three-layer client-edge-cloud* FL system was considered and an optimization algorithm termed HierFAVG was proposed in [21]. Furthermore, an extension to enable multi-step client-edge model aggregation and quantization was studied in [22].

A persistent challenge in FL and distributed optimization in general is that the heterogeneity of local data distributions can significantly harm the optimization performance [14, 18, 38]. Specifically, the computed global model that solves (1) can perform poorly for individual clients if the local data distributions differ significantly. It can also perform poorly when the data ratios of clients in training do not match that of the unseen data in reality, e.g., when the clients process different amounts of data during training due to their differing processing capabilities. To remedy this fairness issue, *minimax* optimization was proposed in [25] as a special case of distributionally robust optimization [11, 30]. In minimax optimization, the weight vector $\boldsymbol{q} = [q_1, \ldots, q_N]$ becomes another set of optimization variables. This new optimization problem is given by

$$\min_{\boldsymbol{w}} \max_{\boldsymbol{q}\in Q} F(\boldsymbol{w}, \boldsymbol{q}), \text{ where } F(\boldsymbol{w}, \boldsymbol{q}) = \sum_{n\in\mathcal{N}} q_n f_n(\boldsymbol{w}), \tag{2}$$

and Q is a subset of the $(N-1)-$dimensional probability simplex. The global model $\boldsymbol{w}$ that solves (2) is guaranteed to be robust against heterogeneity of data distributions. In particular, when constraint Q is inactive, the learned model $\boldsymbol{w}$ performs well even for the worst-off client that has the least favorable local data distribution. The Stochastic Agnostic Federated Learning (Stochastic-AFL) algorithm [25] was proposed to solve the minimax problem (2). It was later extended to accommodate multi-step local updates in Distributionally Robust Federated Averaging (DRFA) [10]. In these methods, for each training round, both the global model $\boldsymbol{w}$ and the weight vector $\boldsymbol{q}$ are updated. However, these works are limited to

ICPP '24, August 12–15, 2024, Gotland, Sweden

ACM ISBN 979-8-4007-1793-2/24/08
https://doi.org/10.1145/3673038.3673137

the conventional two-layer client-server architecture and cannot be applied to the client-edge-cloud network.

This motivates us to design a distributed algorithm to solve the minimax optimization problem (2) over a multi-layer hierarchical network. In particular, we are interest in a solution that has provable performance guarantees. To achieve this objective, we must address several challenges in multiple fronts: 1) The intermediate edge servers have unique impacts on minimax optimization, such as client-edge model aggregation, that has not been explored in existing literature. 2) Minimax optimization requires joint consideration of the mutual impacts of hierarchical model updates and weight updates on algorithm convergence performance. 3) The multi-step local model updates at the clients and multi-step client-edge aggregations can lead to asymmetric synchronization gaps, as each update of the weight vector can only be performed once for multiple steps of model update and aggregation. In this context, the main contributions of this paper are as follows:

- We formulate a constrained minimax optimization problem over the client-edge-cloud architecture. The model parameters and the weight vector for edge areas are jointly optimized to achieve minimax fairness across all edge areas under a general convex constraint. To the best of our knowledge, distributed minimax fair optimization over multi-layer hierarchical networks has not been considered before.
- We propose a distributed algorithm called HierMinimax, which utilizes the client-edge-cloud architecture to reduce the communication overhead via both multi-step local model updates at the clients and multi-step client-edge aggregation. We adopt a checkpoint mechanism, which enables periodic updates of the weight vector for efficient communication. Furthermore, HierMinimax allows partial participation of edge areas, local model updates via local SGD, and most importantly, flexibility of communication frequencies between the clients and the edge servers, as well as between the edge servers and the cloud server.
- We analyze the convergence of HierMinimax to show that it provides performance guarantees for both convex and non-convex loss functions. We explore the communication-convergence trade-off enabled by Hierminimax, showing how we can reduce the communication complexity without overly degrading the convergence rate in distribution minimax optimization. Specifically, for any $\alpha \in [0, 1)$, we can achieve $O(T^{1-\alpha})$ edge-cloud communication complexity with $O(\frac{1}{T^{(1-\alpha)/2}})$ convergence rate for convex loss functions and $O(T^{1-\alpha})$ edge-cloud communication complexity with $O(\frac{1}{T^{(1-\alpha)/4}})$ convergence rate for non-convex loss functions, where T is the number of training time slots.
- For numerical evaluation, we experiment with standard classification datasets for both convex logistic regression and non-convex neural network training. Our experimental results demonstrate that Hierminimax substantially improves the worst-case learning performance and reduces the communication overhead over the current best alternatives.

The rest of this paper is structured as follows. In Section 2, we present the related work. In Section 3, we describe the system model of multi-layer hierarchical networks. We present HierMinimax in Section 4 and its convergence analysis in Section 5. Numerical experiments are provided in Section 6. We conclude the paper in Section 7.

Table 1: Summary of related works on distributed minimax optimization (Hier.: hierarchical, c.c.: communication complexity, c.r.: convergence rate).

		Convex loss		Non-convex loss	
Reference	Hier.	c.c.	c.r.	c.c.	c.r.
[25]	✗	$O(T)$	$O(\frac{1}{T^{1/2}})$	N/A	N/A
[10]	✗	$O(T^{3/4})$	$O(\frac{1}{T^{3/8}})$	$O(T^{3/4})$	$O(\frac{1}{T^{1/8}})$
Ours	✓	$O(T^{1-\alpha})$	$O(\frac{1}{T^{(1-\alpha)/2}})$	$O(T^{1-\alpha})$	$O(\frac{1}{T^{(1-\alpha)/4}})$

2 RELATED WORK

We provide a literature review on hierarchical FL and minimax optimization. The differences between existing works on distributed minimax optimization and our work are summarized in Table 1.

2.1 Hierarchical FL

The standard FL algorithms were designed for two-layer client-server systems [14, 18, 38]. To take advantages of both the connectivity to a large number of clients at the cloud and the availability of low-latency communication at the edge, a three-layer client-edge-cloud FL system was considered in [21, 22]. The HierFAVG algorithm in [21] naturally extends FedAvg to multi-layer networks, and the Hier-Local-QSGD algorithm in [22] further extends HierFAVG with model quantization. A similar study in [5] considered heterogeneous operating rates and showed the dependence of the convergence on the average operating rate, the network topology, and the number of iterations. A convergence bound of hierarchical SGD over non-independent data was derived based on upward and downward divergences in [36]. However, none of these works considered optimizing the model performance over the worst mixture of the data distributions.

2.2 Minimax Optimization

We are interested in learning over multiple distributions via minimax optimization [13, 32] to achieve distributionally robust learning [11, 30]. Minimax optimization has been studied in the game theory and optimization literature [1, 27, 35]. First-order methods for minimax optimization include Gradient Descent Ascent (GDA) [9, 20], Extra-gradient (EG) [16, 24], and Optimistic Gradient Descent Ascent (OGDA) [7]. With the advent and popularity of ML, many ML problems are found to have the minimax structure, such as GANs [12], online learning [31], and adversarial robust learning [33]. However, all of these minimax optimization algorithms require *centralized* implementation.

The authors in [25] were the first to give the formulation of (2) in the context of FL and proposed Stochastic-AFL, which is a stochastic, distributed, and privacy-preserving version of GDA. More recently, in [10], DRFA was proposed to further accommodate a more general communication pattern of FL through periodic model averaging and weight vector updating. Both [10] and [25] are only applicable to conventional two-layer client-server networks. In this work, we study minimax optimization over the client-edge-cloud

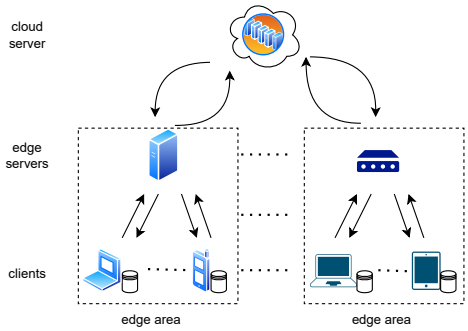

Figure 1: System model of hierarchical networks.

architecture. In particular, HierMinimax differs from DRFA [10] in several key aspects: 1) HierMinimax allows both flexible multi-step local model updates and multi-step client-edge aggregation (see Section 4), 2) HierMinimax is guided by theoretical analysis of the convergence rate and communication complexity caused by hierarchical update and aggregation (see Section 5), and 3) HierMinimax achieves substantial empirical advantages over DRFA (see Section 6).

3 MINIMAX OPTIMIZATION OVER CLIENT-EDGE-CLOUD ARCHITECTURE

We consider a multi-layer hub-and-spoke-type network topology. Since the three-layer client-edge-cloud network architecture is common in practical systems, we use it as an representative example as shown in Fig. 1. All edge servers can directly communicate with the cloud server. The cloud server can indirectly communicate with any client through an edge server.

Let $\mathcal{N}$ be the set of all clients, whose cardinality is $|\mathcal{N}| = N$. Let $\mathcal{E}$ be the set of all edge servers, whose cardinality is $|\mathcal{E}| = N_{\mathcal{E}}$. The set of clients that is associated with edge server $e \in \mathcal{E}$ is denoted by $\mathcal{N}_e$. We define an edge area as an edge server and all clients associated with it. For convenience of notation, we assume the edge servers have the same number of clients, i.e., $|\mathcal{N}_e| = N_0, \forall e \in \mathcal{E}$. Hence, we have $N = N_0 N_{\mathcal{E}}$. Our work can be easily generalized to the case where different edge servers have different numbers of clients.

The clients, edge servers, and cloud server collaborate to compute a global model $\boldsymbol{w}$. We assume the clients within each edge area $e \in \mathcal{E}$ share the same local loss function, denoted by $f_e(w)$. In the FL example, this means that the data samples of clients in each edge area are generated from the same distribution [14, 18, 38]. This is without loss of generality, since if there are multiple distributions in an edge area, we can group clients of the same distribution and the edge server as a virtual edge area. Our goal is to jointly optimize $\boldsymbol{w}$ and the edge weights, denoted by $\boldsymbol{p} = [p_1, \ldots, p_{N_{\mathcal{E}}}]$, by minimizing a worst-case global loss, i.e., for the worst mixture of local loss functions (e.g., data distributions in FL) at the edge. Thus, problem (2) takes the following new form for a three-layer client-edge-cloud network:

$$\min_{\boldsymbol{w}\in\mathcal{W}} \max_{\boldsymbol{p}\in\mathcal{P}} F(\boldsymbol{w}, \boldsymbol{p}), \text{ where } F(\boldsymbol{w}, \boldsymbol{p}) = \sum_{e\in\mathcal{E}} p_e f_e(\boldsymbol{w}), \tag{3}$$

Algorithm 1 HierMinimax Optimization (HierMinimax)

1: Cloud initializes $\boldsymbol{w}^{(0)}$ and $\boldsymbol{p}^{(0)} = [1/N_{\mathcal{E}}, \ldots, 1/N_{\mathcal{E}}]$.
2: **for** each round $k = 0, \ldots, K-1$ **do**
  // Phase 1
3: Cloud samples edge servers $\mathcal{E}^{(k)} \subseteq \mathcal{E}$ by $\boldsymbol{p}^{(k)}$.
4: Cloud samples (c_1, c_2) uniformly from $[\tau_1] \times [\tau_2]$.
5: Cloud broadcasts $\boldsymbol{w}^{(k)}$ and (c_1, c_2) to $\mathcal{E}^{(k)}$.
6: **for** each edge server $e \in \mathcal{E}^{(k)}$ **do**
7: $\boldsymbol{w}_e^{(k,\tau_2)}, \boldsymbol{w}_e^{(k,c_2,c_1)} = \text{ModelUpdate}(\boldsymbol{w}^{(k)}, c_1, c_2)$.
8: Cloud aggregates the global model via (5).
9: Cloud computes the checkpoint model via (6).
  // Phase 2
10: Cloud samples edges $\mathcal{U}^{(k)} \subseteq \mathcal{E}$ uniformly.
11: Cloud broadcasts $\boldsymbol{w}^{(k,c_2,c_1)}$ to $\mathcal{U}^{(k)}$.
12: **for** each edge server $e \in \mathcal{U}^{(k)}$ **do**
13: $f_e(\boldsymbol{w}^{(k,c_2,c_1)}) = \text{LossEstimation}(\boldsymbol{w}^{(k,c_2,c_1)})$.
14: Cloud updates weight vector via (7).

$\mathcal{W} \subseteq \mathbb{R}^d$, and $\mathcal{P} \subseteq \Delta_{N_{\mathcal{E}}-1}$ with $\Delta_{N_{\mathcal{E}}-1}$ being the $(N_{\mathcal{E}}-1)$-dimensional probability simplex. We allow both $\mathcal{W}$ and $\mathcal{P}$ to be any compact convex set.[1]

A naive approach to solve (3) is to directly adopt Stochastic-AFL as follows. For each *training round*, the cloud server broadcasts the current global model to all the clients through the edge servers. Each client updates the model via single-step local SGD and sends it through an edge server to the cloud server for global aggregation. The edge weights are then updated by gradient ascent utilizing the loss estimation of the global model among the edge areas. However, the communication overhead of this approach is high, as communication between the edge areas and the cloud server is required per training round of both the model update and the weight update.

Ideally, a communication-efficient algorithm should allow multi-step local model updates at the clients and multi-step client-edge aggregations per training round. However, this will require a way to evenly sample the local loss values between the aggregation instances over the entire training process, which is needed to update the edge weights. We follow this idea and propose HierMinimax, which we will provide provable performance guarantees.

4 HIERARCHICAL DISTRIBUTED MINIMAX ALGORITHM

The HierMinimax algorithm is carried out over K *training rounds*. Each training round corresponds to one update of the global model parameters $\boldsymbol{w}$ and the edge weights $\boldsymbol{p}$ by the cloud server. In each training round, τ_2 client-edge model aggregations are performed. Each client-edge aggregation in turn is performed after τ_1 local SGD steps at the clients. Thus, the total number of training time slots is $T = K\tau_1\tau_2$.

[1] Note that if $\mathcal{P} = \Delta_{N_{\mathcal{E}}-1}$, the optimization over $\boldsymbol{p}$ is essentially unconstrained and the optimal minimax solution maximizes performance for the worst-off edge area that has the least favorable loss function. Our formulation here is more general, where $\mathcal{P}$ may represent, e.g., prior knowledge or parameter regularization.

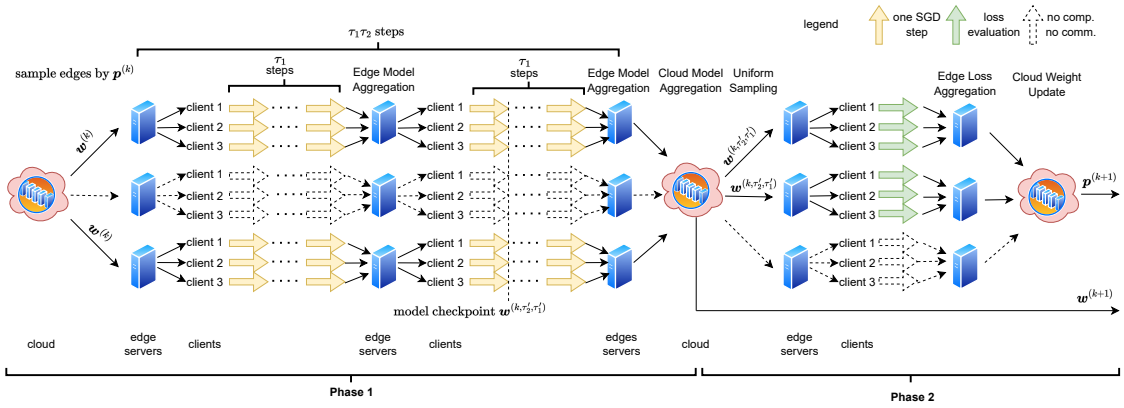

Figure 2: One training round for the updates of model parameters and the weight vector for HierMinimax.

We next discuss the detailed design of HierMinimax, while its formal specification is given in Algorithm 1. As shown in Fig. 2, each training round of HierMinimax contains *two phases*. In Phase 1, the cloud server updates the global model parameters $\boldsymbol{w}$ after τ_2 client-edge model aggregations, which corresponds to $\tau_1\tau_2$ local SGD steps at the clients. In this phase, the algorithm also obtains a random checkpoint of an intermediate model. In Phase 2, the cloud server updates the edge weights $\boldsymbol{p}$ based on the checkpoint model generated in Phase 1.

4.1 Phase 1: Model Parameter Update

In Phase 1 of each training round k, the cloud server first samples $m_{\mathcal{E}}$ edge servers, denoted by a set $\mathcal{E}^{(k)}$, based on the probability defined by the edge weights $\boldsymbol{p}^{(k)}$ of the current round. The cloud server also samples a *checkpoint* index (c_1, c_2) from $[\tau_1] \times [\tau_2]$ uniformly at random. The cloud server then broadcasts the current global model $\boldsymbol{w}^{(k)}$ and the checkpoint index (c_1, c_2) to each sampled edge server. Each sampled edge server $e \in \mathcal{E}^{(k)}$ runs a ModelUpdate procedure based on $\boldsymbol{w}^{(k)}$ and (c_1, c_2) received from the cloud. The ModelUpdate procedure contains two parts.

Part (a): Edge Model Update. Each edge server $e \in \mathcal{E}^{(k)}$ performs τ_2 client-edge model aggregations. At the (t_2+1)-th client-edge aggregation, each edge server e first broadcasts the model $\boldsymbol{w}_e^{(k,t_2)}$, which is the model at the edge server e after t_2 client-edge aggregations in training round k, to all clients $\mathcal{N}_e$ associated with the edge server. Each client $n \in \mathcal{N}_e$ then performs τ_1 steps of local SGD starting from $\boldsymbol{w}_e^{(k,t_2)}$, i.e., at each step $t_1 \in \{0, \ldots, \tau_1 - 1\}$, each client n performs the following SGD update:

$$\boldsymbol{w}_n^{(k,t_2,t_1+1)} = \Pi_{\mathcal{W}}(\boldsymbol{w}_n^{(k,t_2,t_1)} - \eta_w \nabla_w f_n(\boldsymbol{w}_n^{(k,t_2,t_1)}; \xi_n^{(t_1)})), \quad (4)$$

where $\boldsymbol{w}_n^{(k,t_2,t_1)}$ is the local model after t_1 local model update and t_2 client-edge aggregation in k-th training round, η_w is the learning rate on the model update, $\xi_n^{(t_1)}$ is a mini-batch training examples sampled by client n, and $\Pi_{\mathcal{W}}(\cdot)$ is the projection onto set $\mathcal{W}$. After finishing τ_1 steps of local SGD in (4), each client $n \in \mathcal{N}_e$ sends back $\boldsymbol{w}_n^{(k,t_2,\tau_1)}$ to the edge server $e \in \mathcal{E}^{(k)}$. Finally, each edge server $e \in \mathcal{E}^{(k)}$ performs client-edge model aggregation via $\boldsymbol{w}_e^{(k,t_2+1)} = \frac{1}{N_0} \sum_{n \in \mathcal{N}_e} \boldsymbol{w}_n^{(k,t_2,\tau_1)}$.

Part (b): Checkpoint Model Update. Each edge server $e \in \mathcal{E}^{(k)}$ obtains the checkpoint index (c_1, c_2) from the cloud server. It broadcasts c_1 to its clients $\mathcal{N}_e$ at the beginning of the c_2-th step of local aggregation. Then, when each client $n \in \mathcal{N}_e$ sends back its updated model $\boldsymbol{w}_n^{(k,t_2,\tau_1)}$ to edge server $e \in \mathcal{E}^{(k)}$, it also sends along the checkpoint model $\boldsymbol{w}_n^{(k,c_2,c_1)}$. Each edge server $e \in \mathcal{E}^{(k)}$ updates its aggregate checkpoint model via $\boldsymbol{w}_e^{(k,c_2,c_1)} = \frac{1}{N_0} \sum_{n \in \mathcal{N}_e} \boldsymbol{w}_n^{(k,c_2,c_1)}$. The checkpoint model at the edge server will be used to approximate the evolution of the local models for updating $\boldsymbol{p}$ in Phase 2 to guarantee convergence.

After parts (a) and (b) of the ModelUpdate procedure, each sampled edge server $e \in \mathcal{E}^{(k)}$ sends $\boldsymbol{w}_e^{(k,\tau_2)}$ and $\boldsymbol{w}_e^{(k,c_2,c_1)}$ to the cloud server. The cloud server then performs edge-cloud model aggregation to update the global model via

$$\boldsymbol{w}^{(k+1)} = \frac{1}{m_{\mathcal{E}}} \sum_{e \in \mathcal{E}^{(k)}} \boldsymbol{w}_e^{(k,\tau_2)}. \quad (5)$$

The cloud server also updates a new checkpoint model via

$$\boldsymbol{w}^{(k,c_2,c_1)} = \frac{1}{m_{\mathcal{E}}} \sum_{e \in \mathcal{E}^{(k)}} \boldsymbol{w}_e^{(k,c_2,c_1)}. \quad (6)$$

4.2 Phase 2: Edge Weight Update

In each training round k, the cloud server uniformly samples a set of edge servers $\mathcal{U}^{(k)}$ of size $m_{\mathcal{E}}$ and broadcasts the checkpoint model $\boldsymbol{w}^{(k,c_2,c_1)}$ to these edge servers. Note that $\mathcal{U}^{(k)}$ can be different from $\mathcal{E}^{(k)}$ in Phase 1. For each sampled edge server $e \in \mathcal{U}^{(k)}$, it runs the LossEstimation procedure. Specifically, each sampled edge server $e \in \mathcal{U}^{(k)}$ broadcasts the checkpoint model $\boldsymbol{w}^{(k,c_2,c_1)}$ to its clients $\mathcal{N}_e$. Each client $n \in \mathcal{N}_e$ calculates $f_n(\boldsymbol{w}^{(k,c_2,c_1)}; \xi_n^{(k)})$, i.e., the loss on the checkpoint model via a mini-batch of samples $\xi_n^{(k)}$. After receiving the loss calculated from its clients, each edge server $e \in \mathcal{U}^{(k)}$ estimates the loss of the checkpoint model over its data distribution via $f_e(\boldsymbol{w}^{(k,c_2,c_1)}) = \frac{1}{N_0} \sum_{n \in \mathcal{N}_e} f_n(\boldsymbol{w}^{(k,c_2,c_1)}; \xi_n^{(k)})$.

Each sampled edge server $e \in \mathcal{U}^{(k)}$ then sends its estimated loss to the cloud server. After receiving the loss estimations from all the sampled edge servers, the cloud server constructs a vector $\boldsymbol{v} = [v_1, \ldots, v_{N_{\mathcal{E}}}]$ where $v_e = \frac{N_{\mathcal{E}}}{m_{\mathcal{E}}} f_e(\boldsymbol{w}^{(k,c_2,c_1)})$ if $e \in \mathcal{U}^{(k)}$, else

$v_e = 0$. Recall that the minimax optimization objective in (3) is $F(\boldsymbol{w}, \boldsymbol{p}) = \sum_{e\in\mathcal{E}} p_e f_e(\boldsymbol{w})$. The e-th coordinate of its gradient with respect to $\boldsymbol{p}$ is $[\nabla_{\boldsymbol{p}} F(\boldsymbol{w}, \boldsymbol{p})]_e = f_e(\boldsymbol{w}), \forall \boldsymbol{w} \in \mathcal{W}$. The expectation of the e-th coordinate of $\boldsymbol{v}$ is $\mathbb{E}[v_e] = \frac{m_\mathcal{E}}{N_\mathcal{E}} \frac{N_\mathcal{E}}{m_\mathcal{E}} f_e(\boldsymbol{w}^{(k,c_2,c_1)}) + (1 - \frac{m_\mathcal{E}}{N_\mathcal{E}}) \times 0 = f_e(\boldsymbol{w}^{(k,c_2,c_1)})$. Hence, the constructed $\boldsymbol{v}$ is an unbiased estimator of $\nabla_{\boldsymbol{p}} F(\boldsymbol{w}^{(k,c_2,c_1)}, \boldsymbol{p})$. The cloud server then updates the edge weights via projected gradient ascent, given by

$$\boldsymbol{p}^{(k+1)} = \Pi_\mathcal{P}(\boldsymbol{p}^{(k)} + \eta_p \tau_1 \tau_2 \boldsymbol{v}), \tag{7}$$

where η_p is the learning rate on $\boldsymbol{p}$ and $\Pi_\mathcal{P}(\cdot)$ is the projection onto set $\mathcal{P}$.

5 CONVERGENCE ANALYSIS

In this section, we show that HierMinimax provides guaranteed convergence performance for both convex and non-convex local loss functions. We note that new analysis techniques are required to capture the three-layer client-edge-cloud architecture, as well as multi-step local SGD updates and multi-step client-edge aggregations. All proof details are given in the appendix with proof prerequisites given in Appendix A.

We make the following assumptions, which are common in existing works on distributed minimax optimization [10, 20, 25].

Assumption 1 (Bounded Domains). *The diameters of the compact convex sets $\mathcal{W}$ and $\mathcal{P}$ are $R_\mathcal{W}$ and $R_\mathcal{P}$, respectively.*

Assumption 2 (L-smoothness). *There exists some positive L such that $\|\nabla f_n(\boldsymbol{w}_1) - \nabla f_n(\boldsymbol{w}_2)\| \le L\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|, \forall \boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{W}, n \in \mathcal{N}$ and $\|\nabla F(\boldsymbol{w}_1, \boldsymbol{p}_1) - \nabla F(\boldsymbol{w}_2, \boldsymbol{p}_2)\| \le L\|(\boldsymbol{w}_1, \boldsymbol{p}_1) - (\boldsymbol{w}_2, \boldsymbol{p}_2)\|, \forall (\boldsymbol{w}_1, \boldsymbol{p}_1), (\boldsymbol{w}_2, \boldsymbol{p}_2) \in (\mathcal{W}, \mathcal{P})$.*

Assumption 3 (Bounded Gradients). *There exist some positive G_w and G_p such that $\|\nabla_{\boldsymbol{w}} f_n(\boldsymbol{w})\|_2 \le G_w$ and $\|\nabla_{\boldsymbol{p}} F(\boldsymbol{w}, \boldsymbol{p})\|_2 \le G_p$.*

Assumption 4 (Bounded Stochastic Gradient Variance). *There exist some positive σ_w^2 and σ_p^2 such that $\mathbb{E}[\|\nabla f_n(\boldsymbol{w}; \xi) - \nabla f_n(\boldsymbol{w})\|_2^2] \le \sigma_w^2, \forall n \in \mathcal{N}, \boldsymbol{w} \in \mathcal{W}$, and $\mathbb{E}[\|\nabla_{\boldsymbol{p}} F(\boldsymbol{w}, \boldsymbol{p}; \xi) - \nabla_{\boldsymbol{p}} F(\boldsymbol{w}, \boldsymbol{p})\|_2^2] \le \sigma_p^2, \forall \boldsymbol{w} \in \mathcal{W}$ and $\boldsymbol{p} \in \mathcal{P}$.*

Assumption 5 (Bounded Gradient Dissimilarity). *Let $\nabla f_e(\boldsymbol{w})$ be the gradient of the edge loss function $f_e(\boldsymbol{w})$ for any $e \in \mathcal{E}$. There exists some positive Ψ such that the gradient dissimilarity is bounded by $\sup_{\boldsymbol{w}\in\mathcal{W}, \boldsymbol{p}\in\mathcal{P}, e\in\mathcal{E}} \sum_{j\in\mathcal{E}} p_j \|\nabla f_e(\boldsymbol{w}) - \nabla f_j(\boldsymbol{w})\|^2 \le \Psi$.*

5.1 Convex Loss

We first analyze the performance of HierMinimax when the local loss functions are convex in $\boldsymbol{w}$, i.e., $f_n(\boldsymbol{w}_1) \ge f_n(\boldsymbol{w}_2) + \nabla f_n(\boldsymbol{w}_2)^T(\boldsymbol{w}_1 - \boldsymbol{w}_2), \forall \boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{W}, \forall n \in \mathcal{N}$. Then the global loss function $F(\boldsymbol{w}, \boldsymbol{p})$ is convex in $\boldsymbol{w}$. Since $F(\boldsymbol{w}, \boldsymbol{p})$ is also linear in $\boldsymbol{p}$, it is a convex-concave function. The standard way to measure the optimality of a solution $(\hat{\boldsymbol{w}}, \hat{\boldsymbol{p}})$ for constrained convex-concave optimization is the duality gap, given by

$$\max_{\boldsymbol{p}\in\mathcal{P}} F(\hat{\boldsymbol{w}}, \boldsymbol{p}) - \min_{\boldsymbol{w}\in\mathcal{W}} F(\boldsymbol{w}, \hat{\boldsymbol{p}}), \tag{8}$$

where $\hat{\boldsymbol{p}} = \frac{1}{K}\sum_{k=0}^{K-1} \boldsymbol{p}^{(k)}$ is the time-averaged edge weights, $\hat{\boldsymbol{w}} = \frac{1}{mT}\sum_{t=0}^{T-1}\sum_{n\in S^{(t)}} \boldsymbol{w}_n^{(t)}$ is the time-averaged model parameters, and $S^{(t)}$ is the set of sampled clients at iteration t, whose cardinality is $m = m_\mathcal{E} N_0$. Let $(\boldsymbol{w}^*, \boldsymbol{p}^*)$ be a minimax point, in the sense that a Nash equilibrium is established [26]. The duality gap of $(\boldsymbol{w}^*, \boldsymbol{p}^*)$ is zero as $\max_{\boldsymbol{p}\in\mathcal{P}} F(\boldsymbol{w}^*, \boldsymbol{p}) = F(\boldsymbol{w}^*, \boldsymbol{p}^*) = \min_{\boldsymbol{w}\in\mathcal{W}} F(\boldsymbol{w}, \boldsymbol{p}^*)$ for convex-concave functions by the von Neumann minimax theorem [35]. Hence, the lower the duality gap is, the better the solution $(\hat{\boldsymbol{w}}, \hat{\boldsymbol{p}})$ is, as an approximation of a Nash equilibrium.

We require a key lemma in our convergence analysis which bounds the squared distance between any local model $\boldsymbol{w}_n^{(t)}$ and the virtual global model $\boldsymbol{w}^{(t)} = \frac{1}{m}\sum_{n\in S^{(t)}} \boldsymbol{w}_n^{(t)}$.

Lemma 1 (Bounded Squared Model Divergence). *For Algorithm 1 with convex loss, assuming $1 - 20\eta_w^2 L^2 \tau_1^2 (1 + \tau_2^2) \ge \frac{1}{2}$, we have*

$$\frac{1}{mT}\sum_{t=0}^{T-1}\sum_{n\in S^{(t)}} \mathbb{E}\big[\big\|\boldsymbol{w}^{(t)} - \boldsymbol{w}_n^{(t)}\big\|^2\big] \le 20\eta_w^2\tau_1^2\Big(\frac{m+1}{m}\sigma_w^2 + \Psi\Big) + 20\eta_w^2\tau_1^2\tau_2^2\Big(\frac{m_\mathcal{E}+1}{N_0}\sigma_w^2 + \Psi\Big).$$

Proof. See Appendix B. □

Leveraging Lemma 1, we have the following theorem.

Theorem 1. *The duality gap achieved by HierMinimax for convex loss is upper bounded by*

$$\mathbb{E}\Big[\max_{\boldsymbol{p}\in\mathcal{P}} F(\hat{\boldsymbol{w}}, \boldsymbol{p}) - \min_{\boldsymbol{w}\in\mathcal{W}} F(\boldsymbol{w}, \hat{\boldsymbol{p}})\Big] \le \underbrace{\frac{R_\mathcal{P}^2}{2\eta_p T} + \frac{\eta_p\tau_1\tau_2}{2}G_p^2 + \frac{\eta_p\tau_1\tau_2}{2m}\sigma_p^2}_{\text{maximization gap}} + \underbrace{\frac{N_\mathcal{E} R_\mathcal{W}^2}{2\eta_w T} + \frac{\eta_w N_\mathcal{E}}{2}G_w^2 + \frac{\eta_w}{2N_0}\sigma_w^2}_{\text{minimization gap}}$$
$$+ \underbrace{10LN_\mathcal{E}\eta_w^2\tau_1^2\Big(\frac{m+1}{m}\sigma_w^2 + \Psi\Big)}_{\text{client-edge aggregation}} + \underbrace{10LN_\mathcal{E}\eta_w^2\tau_1^2\tau_2^2\Big(\frac{m_\mathcal{E}+1}{N_0}\sigma_w^2 + \Psi\Big)}_{\text{edge-cloud aggregation}}.$$

Proof. We require the following two lemmas: Lemma 3 bounds the update of model $\boldsymbol{w}$ and Lemma 4 bounds the update on weight $\boldsymbol{p}$. The proof details are given in Appendix C. □

Theorem 1 provides a means to tune the tradeoff between the communication complexity and the convergence rate. Suppose we set τ_1 and τ_2 such that $\tau_1\tau_2 \in \Theta(T^\alpha)$ for any $\alpha \in [0, 1)$. The communication complexity between the edge and cloud servers is $\Theta(T^{1-\alpha})$. Let $\eta_p = \Theta(\frac{1}{T^{(1+\alpha)/2}})$, and let $\eta_w = \Theta(\frac{1}{T^{1-2\alpha}})$ if $\alpha \in (0, \frac{1}{4})$ and $\eta_w = \Theta(\frac{1}{T^{1/2}})$ if $\alpha \in [\frac{1}{4}, 1)$. From Theorem 1, the convergence rate of HierMinimax is $O(\frac{1}{T^{(1-\alpha)/2}})$. Note that α is a tunable value. As α increases, the communication complexity $O(T^{1-\alpha})$ decreases but the convergence rate $O(\frac{1}{T^{(1-\alpha)/2}})$ becomes worse, i.e., we can trade convergence rate to reduce communication complexity.

One extreme case is to let $\tau_1 = \tau_2 = 1$. Then, the communication complexity is $O(T)$ and the convergence rate is $O(\frac{1}{T^{1/2}})$. This recovers the same scaling in T for Stochastic-AFL in [25]. Another interesting and more general case is $\tau_2 = 1$. When substituting this into Theorem 1, we recover a duality gap bound that scales in T and τ_1 in the same way as DRFA in [10]. We further note that HierMinimax allows any choice from a wider range of trade-off points beyond these special cases.

5.2 Non-convex Loss

The loss functions of many modern ML applications such as neural networks training are non-convex. A Nash equilibrium may not exist in this setting and the duality gap is no longer a meaningful measure of optimality [20]. We first define $\Phi : \mathcal{W} \to \mathbb{R}$ such that $\Phi(\boldsymbol{w}) = \max_{\boldsymbol{p}\in\mathcal{P}} F(\boldsymbol{w}, \boldsymbol{p})$. Hence, our optimization formulation (3) is equivalent to $\min_{\boldsymbol{w}\in\mathcal{W}} \Phi(\boldsymbol{w})$. Since for any given $\boldsymbol{w} \in \mathcal{W}$, $F(\boldsymbol{w}, \cdot)$ is linear in $\boldsymbol{p}$, $\Phi(\boldsymbol{w})$ can be computed efficiently. However, since Φ itself is non-convex in $\boldsymbol{w}$, the problem of finding the global minimum of Φ is still in general NP-hard. Therefore, in the non-convex optimization literature, the stationary point is commonly used for measuring the optimality of the solutions [4]. However, the function Φ is not necessarily differentiable, making it improper to directly use the gradient $\nabla\Phi$.

We first follow the approach in [8, 20] to define a Moreau envelope to facilitate our convergence analysis. The λ-Moreau envelope of a function Φ with a positive parameter λ is

$$\Phi_\lambda(\boldsymbol{w}) = \min_{\boldsymbol{x}\in\mathcal{W}} \left\{\Phi(\boldsymbol{x}) + (1/2\lambda)\|\boldsymbol{x} - \boldsymbol{w}\|^2\right\}. \tag{9}$$

We choose $\lambda = 1/2L$ such that $\Phi_{1/2L}(\cdot)$ is differentiable [20, Lemma 3.6]. Then the stationarity of the function Φ can be approximately measured by $\|\nabla\Phi_{1/2L}(\cdot)\|$.

We observe that the analysis methods in [8, 20, 29] are for algorithms solving minimax optimization of nonconvex-concave objectives with the same updated frequencies on both sets of optimization variables. Therefore, they are not directly applicable to the analysis of HierMinimax, as the edge weights are updated for one time after both multi-step local client model updates and multi-step client-edge aggregations are performed. As part of the new analysis required to bound the performance of HierMinimax, we first present the following lemma to bound the distance between any local model $\boldsymbol{w}_n^{(t)}$ and the virtual global model $\boldsymbol{w}^{(t)} = \frac{1}{m}\sum_{n\in S^{(t)}} \boldsymbol{w}_n^{(t)}$.

Lemma 2 (Bounded Divergence of Model). *For Algorithm 1 with non-convex loss, assuming $1 - 2\eta_w L\tau_1(1+\tau_2) \geq \frac{1}{2}$, we have*

$$\begin{aligned}&\frac{1}{mT}\sum_{t=0}^{T-1}\frac{1}{m}\sum_{n\in S^{(t)}}\mathbb{E}\left[\left\|\boldsymbol{w}^{(t)} - \boldsymbol{w}_n^{(t)}\right\|\right]\\ &\leq 2\eta_w\tau_1\left(\frac{m+1}{m}\sigma_w + \sqrt{\Psi}\right) + 2\eta_w\tau_1\tau_2\left(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w + \sqrt{\Psi}\right).\end{aligned} \tag{10}$$

Proof. The proof follows similar steps as the proof of Lemma 1, and it is omitted due to space constraint. The difference is that we bound $\mathbb{E}[\|\boldsymbol{w}^{(t)} - \boldsymbol{w}_n^{(t)}\|]$ instead of $\mathbb{E}[\|\boldsymbol{w}^{(t)} - \boldsymbol{w}_n^{(t)}\|^2]$ in Lemma 1. □

Leveraging Lemma 2, we have the following theorem.

Theorem 2. *The time-averaged expectation of the squared norm of the $(1/2L)$-Moreau envelope of Φ, achieved by HierMinimax for non-convex loss, is upper bounded by*

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\Phi_{1/2L}(\boldsymbol{w}^{(t)})\right\|^2\right]\\ &\leq \frac{4\Phi_{1/2L}(\boldsymbol{w}^{(0)})}{\eta_w N_{\mathcal{E}} T} + 16L\sqrt{K}\eta_w\tau_1\tau_2 G_w\sqrt{G_w^2 + \sigma_w^2}\\ &\quad + 4L\frac{R_{\mathcal{P}}^2}{\sqrt{K}\eta_p\tau_1\tau_2} + 8\eta_p\tau_1\tau_2 L(G_p^2 + \frac{\sigma_p^2}{m})\\ &\quad + \frac{4\eta_w}{N_{\mathcal{E}}}(G_w^2 + \frac{\sigma_w^2}{m}) + \frac{8\eta_w\tau_1 R_{\mathcal{W}} L^2}{N_{\mathcal{E}}}(\frac{m+1}{m}\sigma_w + \sqrt{\Psi})\\ &\quad + \frac{8\eta_w\tau_1\tau_2 R_{\mathcal{W}} L^2}{N_{\mathcal{E}}}(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w + \sqrt{\Psi}).\end{aligned} \tag{11}$$

Proof. The proof consists of the following key steps. We first derive a bound for the update of $\boldsymbol{w}^{(t+1)}$ with respect to $\Phi_{1/2L}$ and the LHS of (10) in Lemma 2. Then we apply Lemma 4 to bound one update of $\boldsymbol{p}$ in the proof of Theorem 1. Finally, we use Lemma 2 and the block technique in general non-convex-concave optimization [20], which controls the gap between $\boldsymbol{p}$ and its optimal value at each t, to complete the proof. Further details are omitted due to space constraint. □

Similarly to the convex case, Theorem 2 provides a means to tune the tradeoff between the communication complexity and the convergence rate. Let $\tau_1\tau_2 \in \Theta(T^\alpha)$ for any $\alpha \in [0, 1)$. The communication complexity between the edge and cloud servers is $\Theta(T^{1-\alpha})$. Let $\eta_p = \Theta(\frac{1}{T^{(1+3\alpha)/4}})$ and $\eta_w = \Theta(\frac{1}{T^{(3+\alpha)/4}})$, the convergence rate is $O(\frac{1}{T^{(1-\alpha)/4}})$. As α increases, the communication complexity $O(T^{1-\alpha})$ decreases but the convergence rate $O(\frac{1}{T^{(1-\alpha)/4}})$ becomes worse.

For the special case $\tau_1 = \tau_2 = 1$, HierMinimax has communication complexity $O(T)$ and convergence rate $O(\frac{1}{T^{1/4}})$. We note that this is the same as the best possible convergence rate for *centralized* minimax optimization with nonconvex loss in [20]. HierMinimax can achieve this convergence rate while incurring maximum communication complexity, but it can also be tuned to reduce its convergence speed in exchange for higher communication efficiency. For the other special case $\tau_2 = 1$, it is easy to check from Theorem 2 that HierMinimax recovers a Moreau envelope bound that scales in T and τ_1 in the same way as DRFA in [10].

6 NUMERICAL EXPERIMENTS

In addition to deriving the communication complexity and convergence scaling results shown in the previous section, we further conduct experiments to study the numerical performance of HierMinimax. We use distributed machine learning as an example application. Our experiments are conducted via PyTorch version 2.0.1 [28].

We consider the following benchmarks for performance comparison:

- FedAvg [23]. This is the standard method for FL. It does not use edge servers. It solves the minimization problem (1) via local SGD with multi-step local model update in each training round.
- Stochastic-AFL [25]. See Section 3 for details. This method does not use edge servers. It solves the minimax problem (3) by single-step local model update in each training round.
- DRFA [10]. This method is similar to Stochastic-AFL, except that it solves the minimax problem (3) by multi-step local model update in each training round.
- HierFAVG [21]. This method uses the same three-layer client-edge-cloud architecture as HierMinimax but solves the minimization problem (1).

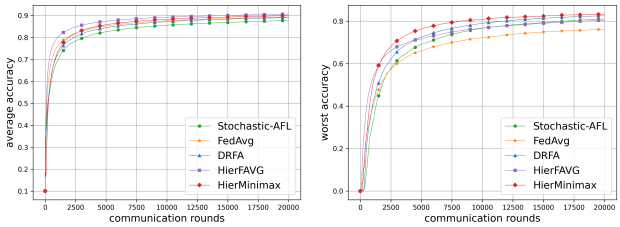

Figure 3: Average and worst test accuracies with convex loss functions (for EMNIST-Digits).

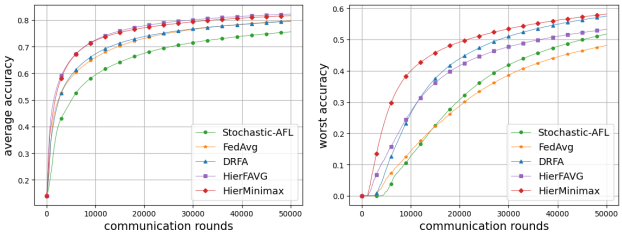

Figure 4: Average and worst test accuracies with non-convex loss functions (for Fashion-MNIST).

For all methods with multi-step model updates, we set $\tau_1 = 2$. Furthermore, for methods utilizing hierarchical architectures, we set $\tau_2 = 2$, i.e., $\tau_1 = \tau_2 = 2$ for both HierFAVG and HierMinimax, so that both models have the same amount of model updates.

6.1 Convex Loss Functions

We consider multinomial logistic regression as the model and cross-entropy loss as the loss function. They are applied to the image classification dataset EMNIST-Digits [6], which contains 10 classes of hand-written digits. We set $N_{\mathcal{E}} = 10$, $N_0 = 3$, $m_{\mathcal{E}} = 5$, $\mathcal{W} = \mathbb{R}^{7850}$, and $\mathcal{P} = \Delta_9$. To create heterogeneous data distributions, we assign one distinct class of training data to the clients of each edge area. We use SGD of learning rate $\eta_w = 0.001$ with batch size 1 for each local model update and learning rate of $\eta_p = 0.001$ for the weight vector update.

Fig. 3 shows a comparison of the average test accuracy and worst test accuracy among the clients. Clearly, the three methods that solve the minimax problem (3) provide much better worst-case performance, while paying only a small price on the average performance. Furthermore, HierMinimax substantially outperforms the two-layer minimax methods. Specifically, to reach 80% worst accuracy, HierMinimax takes only 8200 communication rounds, compared with 16652 rounds for Stochastic-AFL and 11727 rounds for DRFA, corresponding to overhead reduction of 51% and 30%, respectively. HierMinimax also substantially outperforms the three-layer minimization method HierFAVG with communication overhead reduction of 55% from 18228 rounds. We further note that FedAvg does not reach 80% worst accuracy even after 20000 communication rounds.

6.2 Non-convex Loss Functions

For non-convex loss, we consider a two hidden-layer fully-connected neural network, with 300 and 100 neurons in the hidden-layers, ReLU as the activation function, and cross-entropy loss as the loss function. It is applied to perform a more difficult image classification task on Fashion-MNIST [37], which contains 10 classes of clothes. We set $N_{\mathcal{E}} = 10$, $N_0 = 3$, $m_{\mathcal{E}} = 2$, $\mathcal{W} = \mathbb{R}^{266610}$, and $\mathcal{P} = \Delta_9$. To create heterogeneous data distributions, we adopt a more controllable way as in [15]: for s% similarity we allocate to each edge area s% i.i.d. data and the remaining $(100 - s)$% by sorting according to label. Here we present the case $s = 50$. We use SGD of learning rate $\eta_w = 0.001$ with batch size 8 for each local model update and learning rate of $\eta_p = 0.0001$ for the weight vector update.

Fig. 4 shows a comparison of the average test accuracy and worst test accuracy among the clients. Again, the three methods that solve the minimax problem (3) provide substantial improvements on the worst-case performance, compared with their counterparts solving the minimization problem (1), while maintaining similar average performance. HierMinimax again substantially outperforms the two-layer minimax methods and three-layer minimization methods. Specifically, to reach 50% worst accuracy, HierMinimax takes 21576 communication rounds, which is a 52% reduction from the 45201 communication rounds needed by Stochastic-AFL, a 23% reduction from the 28087 communication rounds by DRFA, and a 41% reduction from the 36445 communication rounds by HierFAVG. We further note that FedAvg does not reach 50% worst accuracy even after 50000 communication rounds.

Table 2: Comparison of HierFAVG and HierMinimax.

Datasets	Methods	Average	Worst	Variance
EMNIST-Digits [6]	HierFAVG	**0.9070**	0.8035	21.0504
	HierMinimax	0.8999	**0.8348**	**5.5657**
Fashion-MNIST [37]	HierFAVG	**0.8072**	0.4829	206.6945
	HierMinimax	0.7631	**0.6051**	**24.7095**
MNIST [17]	HierFAVG	**0.8703**	0.7572	30.9331
	HierMinimax	0.8501	**0.7818**	**20.2926**
Adult [2]	HierFAVG	**0.8180**	0.6453	76.2957
	HierMinimax	0.8123	**0.7323**	**16.3589**
Synthetic [19]	HierFAVG	**0.7539**	0.2102	732.6033
	HierMinimax	0.7250	**0.2896**	**478.8593**

6.3 Minimax Fairness and Variance

We further investigate the fairness of HierMinimax in terms of test accuracies achieved by different edge areas. In addition to the average and worst test accuracy discussed earlier, we further compare the variance of test accuracies over edge areas to explicitly consider the uniformity of test accuracies. The first part of Table 2 contains the results of logistic regression models solving image classification problems on EMNIST-Digits [6], Fashion-MNIST [37], and MNIST [17]. We observe that HierMinimax typically yields only slightly lower average test accuracy compared with HierFAVG. However, HierMinimax achieves much higher worst test accuracy, especially for a harder dataset such as Fashion-MNIST. Meanwhile, the variance of test accuracies over different edge areas achieved by HierMinimax is much lower than that of HierFAVG, as much as one order of magnitude in the case of Fashion-MNIST.

Table 2 additionally shows the results for salary prediction on the Adult dataset [2] and sample classification on the Synthetic dataset [19]. For Adult, we consider 2 edge areas, each with data samples from Doctorate and non-Doctorate groups, respectively. We train a logistic regression model on categorical features with

$\eta_w = 0.001$ and $\eta_p = 0.0001$. For Synthetic, we consider 100 edge areas and train a logistic regression model with $\eta_w = \eta_p = 0.0001$. Similar to [19], we report the worst 10% accuracy. We again observe substantial advantage of HierMinimax over HierFAVG on both Adult and Synthetic.

7 CONCLUSION

We propose HierMinimax for distributed minimax optimization over a three-layer network architecture exemplified by the common client-edge-cloud system in mobile edge computing. Our convergence analysis, for both convex and non-convex loss functions, sheds light on the tradeoff between the communication complexity and the convergence rate. In experiments with standard ML datasets, we further show that HierMinimax has substantial advantages in terms of communication overhead reduction and worst-case performance, compared with the existing two-layer approaches and the three-layer minimization approach.

ACKNOWLEDGMENTS

This work was supported by Ericsson, the Natural Sciences and Engineering Research Council of Canada (NSERC), and Mitacs.

REFERENCES

[1] Tamer Basar and Geert Jan Olsder. 1998. *Dynamic Noncooperative Game Theory*. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9781611971132
[2] Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository.
[3] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, et al. 2019. Towards federated learning at scale: System design. In *Proc. of the SysML Conf.*
[4] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. 2018. Optimization methods for large-scale machine learning. *SIAM Rev.* 60, 2 (2018), 223–311.
[5] Timothy Castiglia, Anirban Das, and Stacy Patterson. 2021. Multi-level local SGD: Distributed SGD for heterogeneous hierarchical networks. In *Proc. of ICLR.*
[6] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. In *Proc. of IJCNN.*
[7] Constantinos Daskalakis and Ioannis Panageas. 2018. The limit points of (optimistic) gradient descent in min-max optimization. In *Proc. of NeurIPS.*
[8] Damek Davis and Dmitriy Drusvyatskiy. 2019. Stochastic model-based minimization of weakly convex functions. *SIAM J. on Optim.* 29, 1 (2019), 207–239.
[9] V.F. Dem'yanov and A.B. Pevnyi. 1972. Numerical methods for finding saddle points. *Comput. Math. and Math. Phys.* 12, 5 (1972), 11–52.
[10] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. 2020. Distributionally robust federated averaging. In *Proc. of NeurIPS.*
[11] John C. Duchi and Hongseok Namkoong. 2021. Learning models with uniform performance via distributionally robust optimization. *The Ann. of Statist.* 49, 3 (2021), 1378 – 1406.
[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, et al. 2014. Generative adversarial nets. In *Proc. of NeurIPS.*
[13] Nika Haghtalab, Michael Jordan, and Eric Zhao. 2022. On-demand sampling: Learning optimally from multiple distributions. In *Proc. of NeurIPS.*
[14] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, et al. 2021. Advances and open problems in federated learning. *Found. and Trends in Mach. Learn.* 14, 1-2 (2021), 1–210.
[15] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, et al. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proc. of ICML.*
[16] Galina M Korpelevich. 1976. The extragradient method for finding saddle points and other problems. *Matecon* 12 (1976), 747–756.
[17] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* (2010).
[18] Tian Li, Anit Kumar Sahu, Ameet S. Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.* 37 (2020), 50–60.
[19] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. 2020. Fair resource allocation in federated learning. In *Proc. of ICLR.*
[20] Tianyi Lin, Chi Jin, and Michael Jordan. 2020. On gradient descent ascent for nonconvex-concave minimax problems. In *Proc. of ICML.*
[21] Lumin Liu, Jun Zhang, SH Song, and Khaled B Letaief. 2020. Client-edge-cloud hierarchical federated learning. In *Proc. of ICC.*
[22] Lumin Liu, Jun Zhang, Shenghui Song, and Khaled B Letaief. 2023. Hierarchical federated learning with quantization: Convergence analysis and system design. *IEEE Trans. on Wireless Commun.* 22, 1 (2023), 2–18.
[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proc. of AISTATS.*
[24] Panayotis Mertikopoulos, Bruno Lecouat, Houssam Zenati, Chuan-Sheng Foo, Vijay Chandrasekhar, et al. 2019. Optimistic mirror descent in saddle-point problems: Going the extra(-gradient) mile. In *Proc. of ICLR.*
[25] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In *Proc. of ICML.*
[26] John Nash. 1951. Non-cooperative games. *Ann. of Math.* (1951), 286–295.
[27] Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani. 2007. *Algorithmic game theory.* Cambridge University Press.
[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Proc. of NeurIPS.*
[29] Hassan Rafique, Mingrui Liu, Qihang Lin, and Tianbao Yang. 2022. Weakly-convex–concave min–max optimization: Provable algorithms and applications in machine learning. *Optim. Met. and Softw.* 37, 3 (2022), 1087–1121.
[30] Hamed Rahimian and Sanjay Mehrotra. 2019. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659* (2019).
[31] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. 2015. Online learning via sequential complexities. *J. of Mach. Learn. Res.* 16, 6 (2015), 155–186.
[32] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks. In *Proc. of ICLR.*
[33] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. 2018. Adversarially robust generalization requires more data. In *Proc. of NeurIPS.*
[34] Sebastian U Stich. 2019. Local SGD converges fast and communicates little. In *Proc. of ICLR.*
[35] John von Neumann. 1928. Zur theorie der gesellschaftsspiele. 119, 1 (1928), 295–320.
[36] Jiayi Wang, Shiqiang Wang, Rong-Rong Chen, and Mingyue Ji. 2022. Demystifying why local aggregation helps: Convergence analysis of hierarchical SGD. In *Proc. of AAAI.*
[37] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. https://github.com/zalandoresearch/fashion-mnist.
[38] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Trans. on Intell. Syst. and Technol.* 10, 2 (jan 2019), 1–19.

A PREREQUISITES TO PROOFS

We first introduce the symbols and notations that will be used throughout our appendices. We use $k \in [K]$ to indicate the k-th training round and $t \in [T]$ to indicate the t-th training time slots (iterations) with $T = K\tau_1\tau_2$. Let the sampled clients after k-th global aggregation be $S^{(k)} = \cup_{e\in\mathcal{E}^{(k)}} \mathcal{N}_e$. We denote the cardinality of $S^{(k)}$ by $m = m_{\mathcal{E}} N_0$, where $m_{\mathcal{E}}$ is the number of sampled edge servers and N_0 is the number of clients in each edge area. Note that for any $t \in [T]$, we have $S^{(t)} = S^{(k)}$ for any $k\tau_1\tau_2 \le t < (k+1)\tau_1\tau_2$.

To facilitate our proofs, we define the following auxiliary variables. At iteration t, let the average model of all selected clients be $\boldsymbol{w}^{(t)} = \frac{1}{m}\sum_{n\in S^{(t)}} \boldsymbol{w}_n^{(t)}$. The full gradient of the model $\boldsymbol{w}_n^{(t)}$ at client n is denoted by $g_n^{(t)} = \nabla f_n(\boldsymbol{w}_n^{(t)})$ and the stochastic gradient of the model $\boldsymbol{w}_n^{(t)}$ at client n is denoted by $h_n^{(t)} = \nabla f_n(\boldsymbol{w}_n^{(t)}; \xi_n^{(t)})$. The average full and stochastic gradient at edge server e are denoted by $g_e^{(t)} = \frac{1}{N_0}\sum_{n\in\mathcal{N}_e} g_n^{(t)}, h_e^{(t)} = \frac{1}{N_0}\sum_{n\in\mathcal{N}_e} h_n^{(t)}$. The average full and stochastic gradient of all selected clients are denoted by $g^{(t)} = \frac{1}{m}\sum_{n\in S^{(t)}} g_n^{(t)}, h^{(t)} = \frac{1}{m}\sum_{n\in S^{(t)}} h_n^{(t)}$.

Recall that at iteration t, the weight vector $\boldsymbol{p}$ has been updated $\lfloor \frac{t}{\tau_1\tau_2} \rfloor$ times, so we denote the global minimax objective at time t by $F(\boldsymbol{w}^{(t)}, \boldsymbol{p}^{(\lfloor \frac{t}{\tau_1\tau_2} \rfloor)})$. We note that the stochastic gradient $h^{(t)}$

is an unbiased estimate of the full gradient of $F(\boldsymbol{w}^{(t)}, \boldsymbol{p}^{(\lfloor \frac{t}{\tau_1\tau_2} \rfloor)})$, i.e., $\mathbb{E}_{\mathcal{E}^{(t)},\{\xi_n^{(t)}\}_{n\in S^{(t)}}}[h^{(t)}] = \mathbb{E}_{\mathcal{E}^{(t)}}[g^{(t)}] = \mathbb{E}[\sum_{e\in\mathcal{E}} p_e^{(\lfloor \frac{t}{\tau_1\tau_2} \rfloor)} \nabla f_e(\boldsymbol{w}_e^{(t)})]$. Furthermore, from Assumption 4, we can show that the variance of $h^{(t)}$ is bounded by $\mathbb{E}\big[\|h^{(t)} - g^{(t)}\|^2\big] \le \frac{\sigma_w^2}{m}$.

Now, we derive the bias and variance of the estimation of $\boldsymbol{p}$. Recall that $\boldsymbol{v}^{(k)}$ is the vector constructed in Phase 2 of training round k. Denote the stochastic gradient of $\boldsymbol{p}$ in this training round by $\boldsymbol{u}^{(k)} = \tau_1\tau_2\boldsymbol{v}^{(k)}$. The full gradient of $F(\boldsymbol{w}^{(t)}, \boldsymbol{p})$ with respect to $\boldsymbol{p}$ is $\bar{\boldsymbol{v}}^{(t)}$. Its e-th coordinate is $[\bar{\boldsymbol{v}}^{(t)}]_e = f_e(\boldsymbol{w}^{(t)}) = [\nabla_p F(\boldsymbol{w}^{(t)}, \boldsymbol{p})]_e$ for all $e \in \mathcal{E}$. We further define $\bar{\boldsymbol{u}}^{(k)} = \sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2} \bar{\boldsymbol{v}}^{(t)}$, which is the summation of the virtual full gradient with respect to $\boldsymbol{p}$ over $\tau_1\tau_2$ iterations within the k-th round. Then the stochastic gradient of $\boldsymbol{p}$ in our algorithm is unbiased, i.e., $\mathbb{E}\left[\boldsymbol{u}^{(k)}\right] = \mathbb{E}\left[\eta_p\tau_1\tau_2\boldsymbol{v}^{(k)}\right] = \bar{\boldsymbol{u}}^{(k)}$, as we sample a timestamp from an interval of length $\tau_1\tau_2$ and sample each edge server uniformly and independently. Note that $\boldsymbol{u}^{(k)}$ also has bounded variance $\mathbb{E}\big[\big\|\boldsymbol{u}^{(k)} - \bar{\boldsymbol{u}}^{(k)}\big\|^2\big] \le \frac{\tau_1^2\tau_2^2}{m}\sigma_p^2$.

B PROOF OF LEMMA 1

Proof. We first bound each $A^{(t)}$ for $t \in [k\tau_1\tau_2, (k+1)\tau_1\tau_2]$, i.e., $t = k\tau_1\tau_2 + t_2\tau_1 + t_2$ where $0 \le t_2 < \tau_2$ and $0 \le t_1 \le \tau_1$. Let e be the edge server with which client n is associated. We define

$$\mathrm{C1} = \sum_{i=0}^{t_2-1} \frac{1}{N_0} \sum_{n'\in\mathcal{N}_e} \sum_{j=0}^{\tau_1-1} \eta_w h_{n'}^{(k,i,j)} \text{ and } \mathrm{C2} = \sum_{j=0}^{t_1-1} \eta_w h_n^{(k,t_2,j)}.$$

We have $\boldsymbol{w}_n^{(t)} = \boldsymbol{w}^{(k,0,0)} - \mathrm{C1} - \mathrm{C2}$. We further define

$$\mathrm{C3} = \frac{1}{m_{\mathcal{E}}} \sum_{e'\in\mathcal{E}^{(t)}} \sum_{i=0}^{t_2-1} \frac{1}{N_0} \sum_{n''\in\mathcal{N}_{e'}} \sum_{j=0}^{\tau_1-1} \eta_w h_{n''}^{(k,i,j)}$$

$$\mathrm{C4} = \frac{1}{m} \sum_{n'\in S^{(t)}} \sum_{j=0}^{t_1-1} \eta_w h_n^{(k,t_2,j)}.$$

We have $\boldsymbol{w}^{(t)} = \boldsymbol{w}^{(k,0,0)} - \mathrm{C3} - \mathrm{C4}$.

We define $A^{(t)} = \frac{1}{m}\sum_{n\in S^{(t)}} \mathbb{E}[\|\boldsymbol{w}_n^{(t)} - \boldsymbol{w}^{(t)}\|^2]$. Hence,

$$A^{(t)} \overset{(a)}{\le} \frac{2}{m} \sum_{n\in S^{(t)}} \mathbb{E}\big[\big\|\mathrm{C1} - \mathrm{C3}\big\|^2\big] + \frac{2}{m} \sum_{n\in S^{(t)}} \mathbb{E}\big[\big\|\mathrm{C2} - \mathrm{C4}\big\|^2\big], \quad (12)$$

where (a) follows from $\|\boldsymbol{x}+\boldsymbol{y}\|^2 \le 2(\|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2)$.

We now bound the right-hand side (RHS) of (12). For the first term, we have

$$\begin{aligned}
&\frac{2}{m} \sum_{n\in S^{(t)}} \mathbb{E}\big[\big\|\mathrm{C1} - \mathrm{C3}\big\|^2\big] \\
&\overset{(a)}{\le} \frac{2\eta_w^2 t_2\tau_1}{m} \sum_{n\in S^{(t)}} \sum_{i=0}^{t_2-1} \sum_{j=0}^{\tau_1-1} \mathbb{E}\Big[\Big\| \sum_{n'\in\mathcal{N}_e} \frac{h_{n'}^{(k,i,j)}}{N_0} - \sum_{n''\in S^{(t)}} \frac{h_{n''}^{(k,i,j)}}{m} \Big\|^2\Big] \\
&\overset{(b)}{\le} 10\eta_w^2 t_2\tau_1 \sum_{i=0}^{t_2-1} \sum_{j=0}^{\tau_1-1} \Big(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w^2 + 2L^2 A^{(k,i,j)} + \Psi\Big) \\
&\overset{(c)}{\le} 10\eta_w^2 \tau_2\tau_1 \sum_{i=0}^{\tau_2-1} \sum_{j=0}^{\tau_1-1} \Big(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w^2 + 2L^2 A^{(k,i,j)} + \Psi\Big), \qquad (13)
\end{aligned}$$

where (a) is by Jensen's inequality; (b) is by first both adding and subtracting four terms $\frac{1}{N_0}\sum_{n'\in\mathcal{N}_e} g_{n'}^{(k,i,j)}$, $\frac{1}{N_0}\sum_{n'\in\mathcal{N}_e} \nabla f_{n'}(\boldsymbol{w}^{(k,i,j)})$, $\frac{1}{m}\sum_{n''\in S^{(t)}} \nabla f_{n''}(\boldsymbol{w}^{(k,i,j)})$, and $\frac{1}{m}\sum_{n''\in S^{(t)}} g_{n''}^{(k,i,j)}$, and then applying Jensen's inequality, along with Assumptions 2, 4, and 5; and (c) is by upper bounding t_2 by τ_2. Applying the same techniques to the second term on the RHS of (12), we can show that

$$\begin{aligned}
&\frac{2}{m} \sum_{n\in S^{(t)}} \mathbb{E}\big[\big\|\mathrm{C2} - \mathrm{C4}\big\|^2\big] \\
&\le 10\eta_w^2\tau_1 \sum_{j=0}^{\tau_1-1} \Big(\frac{m+1}{m}\sigma_w^2 + 2L^2 A^{(k,t_2,j)} + \Psi\Big). \qquad (14)
\end{aligned}$$

Substituting (13) and (14) into (12) and summing the resulting inequality over t from $k\tau_1\tau_2$ to $(k+1)\tau_1\tau_2 - 1$, we have

$$\begin{aligned}
&\sum_{t_2=0}^{\tau_2-1} \sum_{t_1=0}^{\tau_1-1} A^{(k,t_2,t_1)} \\
&= 10\eta_w^2\tau_1^2 \sum_{t_2=0}^{\tau_2-1} \sum_{j=0}^{\tau_1-1} \Big(\frac{m+1}{m}\sigma_w^2 + 2L^2 A^{(k,t_2,j)} + \Psi\Big) \\
&\quad + 10\eta_w^2\tau_1^2\tau_2^2 \sum_{i=0}^{\tau_2-1} \sum_{j=0}^{\tau_1-1} \Big(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w^2 + 2L^2 A^{(k,i,j)} + \Psi\Big) \\
&\overset{(a)}{\le} 20\eta_w^2\tau_1^3\tau_2\Big(\frac{m+1}{m}\sigma_w^2 + \Psi\Big) + 20\eta_w^2\tau_1^3\tau_2^3\Big(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w^2 + \Psi\Big), \quad (15)
\end{aligned}$$

where (a) is by assuming $1 - 20\eta_w^2 L^2\tau_1^2(1+\tau_2^2) \ge \frac{1}{2}$. Summing (15) over k from 0 to $K-1$ and diving both sides by $T = K\tau_1\tau_2$, we complete the proof. □

C PROOF OF THEOREM 1

We first prove Lemma 3 and Lemma 4 to facilitate the proof of Theorem 1.

Lemma 3 (Upper Bound for One Iteration of $\boldsymbol{w}$). *For all $w \in \mathcal{W}$ and $t \in [T]$, we have*

$$\begin{aligned}
&\mathbb{E}\big[\big\|\boldsymbol{w}^{(t+1)} - \boldsymbol{w}\big\|^2\big] - \mathbb{E}\big[\big\|\boldsymbol{w}^{(t)} - \boldsymbol{w}\big\|^2\big] \le L\eta_w A^{(t)} + \eta_w^2 G_w^2 + \frac{\eta_w^2}{m}\sigma_w^2 \\
&\quad - \frac{2\eta_w}{N_{\mathcal{E}}} \mathbb{E}\big[F(\boldsymbol{w}^{(t)}, \boldsymbol{p}^{(\lfloor \frac{t}{\tau_1\tau_2} \rfloor)}) - F(\boldsymbol{w}, \boldsymbol{p}^{(\lfloor \frac{t}{\tau_1\tau_2} \rfloor)})\big]. \qquad (16)
\end{aligned}$$

Proof. We have

$$\begin{aligned}
\mathbb{E}\big[\big\|\boldsymbol{w}^{(t+1)} - \boldsymbol{w}\big\|^2\big] &\overset{(a)}{\le} \mathbb{E}\big[\big\|\boldsymbol{w}^{(t)} - \eta_w h^{(t)} - \boldsymbol{w}\big\|^2\big] \\
&\overset{(b)}{=} \mathbb{E}\big[\big\|\boldsymbol{w}^{(t)} - \eta_w g^{(t)} - \boldsymbol{w}\big\|^2\big] + \eta_w^2 \mathbb{E}\big[\big\|g^{(t)} - h^{(t)}\big\|^2\big] \\
&\overset{(c)}{\le} \mathbb{E}\big[\big\|\boldsymbol{w}^{(t)} - \boldsymbol{w}\big\|^2\big] + A_1^{(t)} + A_2^{(t)} + \frac{\eta_w^2}{m}\sigma_w^2, \qquad (17)
\end{aligned}$$

where (a) follows from the update rule in (4) and projection onto a closed convex set, (b) follows from both adding and subtracting $\eta_w g^{(t)}$, expanding the squares and eliminating the zero-valued cross-term, and (c) is from bounded variance of $h^{(t)}$ and definitions of $A_2^{(t)} = \eta_w^2 \mathbb{E}[\|g^{(t)}\|^2]$ and $A_1^{(t)} = -2\eta_w \mathbb{E}[(g^{(t)})^T(\boldsymbol{w}^{(t)} - \boldsymbol{w})]$.

We now bound $A_1^{(t)}$ on the RHS of (17). We have $A_1^{(t)}$

$$A_1^{(t)} \overset{(a)}{=} -\frac{2\eta_w}{m} \mathbb{E}\Big[\sum_{e\in\mathcal{E}^{(t)}} \sum_{n\in\mathcal{N}_e} \big(\nabla f_n(\boldsymbol{w}_n^{(t)})\big)^T (\boldsymbol{w}^{(t)} - \boldsymbol{w}_n^{(t)})\Big]$$

$$
\begin{aligned}
&- \frac{2\eta_w}{m}\mathbb{E}\Big[\sum_{e\in\mathcal{E}^{(t)}}\sum_{n\in\mathcal{N}_e}\big(\nabla f_n(\boldsymbol{w}_n^{(t)})\big)^T(\boldsymbol{w}_n^{(t)}-\boldsymbol{w})\Big]\\
&\overset{(b)}{\leq} \frac{2\eta_w}{m}\mathbb{E}\Big[\sum_{e\in\mathcal{E}^{(t)}}\sum_{n\in\mathcal{N}_e} f_n(\boldsymbol{w}_n^{(t)})-f_n(\boldsymbol{w}^{(t)})+\frac{L}{2}\|\boldsymbol{w}^{(t)}-\boldsymbol{w}_n^{(t)}\|^2\Big]\\
&+\frac{2\eta_w}{m}\mathbb{E}\Big[\sum_{e\in\mathcal{E}^{(t)}}\sum_{n\in\mathcal{N}_e} f_n(\boldsymbol{w})-f_n(\boldsymbol{w}_n^{(t)})\Big]\\
&\overset{(c)}{=} -\frac{2\eta_w}{N_{\mathcal{E}}}\mathbb{E}\big[F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})-F(\boldsymbol{w},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})\big]+\eta_w L A^{(t)}, \quad (18)
\end{aligned}
$$

where (a) follows from the definition of $g^{(t)}$, (b) is from the L-smoothness and convexity of f_n for any $n\in\mathcal{N}$, and (c) holds because $f_e=\frac{1}{N_0}\sum_{n\in\mathcal{N}_e} f_n$ and we sample the edge servers based on $\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)}$.

We then bound $A_2^{(t)}$ on the RHS of (17). We have

$$
A_2^{(t)} \overset{(a)}{\leq} \frac{\eta_w^2}{m}\sum_{e\in\mathcal{E}^{(t)}}\sum_{n\in\mathcal{N}_e}\mathbb{E}\big[\big\|\nabla f_n(\boldsymbol{w}_n^{(t)})\big\|^2\big] \overset{(b)}{\leq} \eta_w^2 G_w^2, \quad (19)
$$

where (a) is from Jensen's inequality and (b) is from Assumption 3. Substituting (18) and (19) into (17), we obtain (16). □

Lemma 4 (Upper Bound for One Update of $\boldsymbol{p}$). *For all $\boldsymbol{p}\in\mathcal{P}$ and $k\in[K]$, we have*

$$
\begin{aligned}
&\mathbb{E}\big[\big\|\boldsymbol{p}^{(k+1)}-\boldsymbol{p}\big\|^2\big]-\mathbb{E}\big[\big\|\boldsymbol{p}^{(k)}-\boldsymbol{p}\big\|^2\big]\leq \eta_p^2\tau_1^2\tau_2^2G_p^2+\frac{\eta_p^2\tau_1^2\tau_2^2}{m}\sigma_p^2\\
&-2\eta_p\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}\mathbb{E}\big[F(\boldsymbol{w}^{(t)},\boldsymbol{p})-F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(k)})\big]. \quad (20)
\end{aligned}
$$

Proof. Following similar arguments of Lemma 3, we have

$$
\mathbb{E}\big[\big\|\boldsymbol{p}^{(k+1)}-\boldsymbol{p}\big\|^2\big]\leq\mathbb{E}\big[\big\|\boldsymbol{p}^{(k)}-\boldsymbol{p}\big\|^2\big]+B_1^{(k)}+B_2^{(k)}+\frac{\eta_p^2\tau_1^2\tau_2^2}{m}\sigma_p^2, \quad (21)
$$

where we define $B_1^{(k)}=\mathbb{E}\big[2(\eta_p\bar{u}^{(k)})^T(\boldsymbol{p}^{(k)}-\boldsymbol{p})\big]$ and $B_2^{(k)}=\mathbb{E}\big[\big\|\eta_p\bar{u}^{(k)}\big\|^2\big]$.

We now bound the RHS of (21). By substituting $\bar{u}^{(k)}=\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}\bar{v}^{(t)}$ into $B_1^{(k)}$, we have

$$
\begin{aligned}
B_1^{(k)} &\overset{(a)}{=} 2\eta_p\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}\mathbb{E}\big[\big(\nabla_{\boldsymbol{p}}F(\boldsymbol{w}^{(t)},\boldsymbol{p})\big)^T\big(\boldsymbol{p}^{(k)}-\boldsymbol{p}\big)\big]\\
&\overset{(b)}{=} -2\eta_p\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}\mathbb{E}\big[F(\boldsymbol{w}^{(t)},\boldsymbol{p})-F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(k)})\big], \quad (22)
\end{aligned}
$$

where (a) follows from the definition of $\nabla_{\boldsymbol{p}}F(\boldsymbol{w},\boldsymbol{p})$ and the linearity of expectation, and (b) is from the linearity of $F(\boldsymbol{w},\boldsymbol{p})$ in $\boldsymbol{p}$ given any $\boldsymbol{w}$.

Substituting $\bar{u}^{(k)}$ into $B_2^{(k)}$, we have

$$
B_2^{(k)} \overset{(a)}{\leq} \eta_p^2\tau_1\tau_2\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}\mathbb{E}\big[\big\|\bar{v}^{(t)}\big\|^2\big] \overset{(b)}{\leq} \eta_p^2\tau_1^2\tau_2^2G_p^2, \quad (23)
$$

where (a) is from Jensen's inequality and (b) is from Assumption 3. Substituting (22) and (23) into (21), we prove (20). □

We now prove Theorem 1.

Proof. We start from bounding the expected duality gap

$$
\begin{aligned}
&\max_{\boldsymbol{p}\in\mathcal{P}}\mathbb{E}\big[F(\hat{\boldsymbol{w}},\boldsymbol{p})\big]-\min_{\boldsymbol{w}\in\mathcal{W}}\mathbb{E}\big[F(\boldsymbol{w},\hat{\boldsymbol{p}})\big]\\
&=\max_{\boldsymbol{w}\in\mathcal{W},\boldsymbol{p}\in\mathcal{P}}\big(\mathbb{E}\big[F(\hat{\boldsymbol{w}},\boldsymbol{p})\big]-\mathbb{E}\big[F(\boldsymbol{w},\hat{\boldsymbol{p}})\big]\big)\\
&\overset{(a)}{\leq}\max_{\boldsymbol{w}\in\mathcal{W},\boldsymbol{p}\in\mathcal{P}}\mathbb{E}\Big[\frac{1}{T}\sum_{t=1}^{T}\big(F(\boldsymbol{w}^{(t)},\boldsymbol{p})-F(\boldsymbol{w},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})\big)\Big]\\
&\overset{(b)}{=}\max_{\boldsymbol{w}\in\mathcal{W},\boldsymbol{p}\in\mathcal{P}}(\mathrm{C5}+\mathrm{C6}), \quad (24)
\end{aligned}
$$

where (a) is from the convexity in $\boldsymbol{w}$ and linearity in $\boldsymbol{p}$ and (b) is from the definitions of

$$
\begin{aligned}
\mathrm{C5}&=\mathbb{E}\Big[\frac{1}{T}\sum_{k=0}^{K-1}\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}F(\boldsymbol{w}^{(t)},\boldsymbol{p})-F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(k)})\Big]\\
\mathrm{C6}&=\mathbb{E}\Big[\frac{1}{T}\sum_{t=1}^{T}F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})-F(\boldsymbol{w},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})\Big].
\end{aligned}
$$

We now bound C5. Rearranging the terms of (20) in Lemma 4 and dividing both sides by $\frac{1}{2\eta_p}$, we have

$$
\begin{aligned}
&\sum_{t=k\tau_1\tau_2+1}^{(k+1)\tau_1\tau_2}\mathbb{E}\big[F(\boldsymbol{w}^{(t)},\boldsymbol{p})-F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(k)})\big]\\
&\leq\frac{1}{2\eta_p}\big(\mathbb{E}\big[\big\|\boldsymbol{p}^{(k)}-\boldsymbol{p}\big\|^2\big]-\mathbb{E}\big[\big\|\boldsymbol{p}^{(k+1)}-\boldsymbol{p}\big\|^2\big]\big)\\
&\quad+\frac{\eta_p\tau_1^2\tau_2^2}{2}G_p^2+\frac{\eta_p\tau_1^2\tau_2^2}{2m}\sigma_p^2. \quad (25)
\end{aligned}
$$

Summing (25) over k and dividing both sides by T, we have

$$
\begin{aligned}
\mathrm{C5}&\leq\frac{1}{2\eta_pT}\mathbb{E}\big[\big\|\boldsymbol{p}^{(0)}-\boldsymbol{p}\big\|^2\big]+\frac{\eta_p\tau_1\tau_2}{2}G_p^2+\frac{\eta_p\tau_1\tau_2}{2m}\sigma_p^2\\
&\leq\frac{R_{\mathcal{P}}^2}{2\eta_pT}+\frac{\eta_p\tau_1\tau_2}{2}G_p^2+\frac{\eta_p\tau_1\tau_2}{2m}\sigma_p^2. \quad (26)
\end{aligned}
$$

Next, we bound C6. Rearranging the terms of (16) in Lemma 3 and multiplying both sides by $\frac{N_{\mathcal{E}}}{2\eta_w}$, we have

$$
\begin{aligned}
&\mathbb{E}\big[F(\boldsymbol{w}^{(t)},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})-F(\boldsymbol{w},\boldsymbol{p}^{(\lfloor\frac{t}{\tau_1\tau_2}\rfloor)})\big]\leq\frac{\eta_wN_{\mathcal{E}}}{2}G_w^2+\frac{\eta_w^2}{N_0}\sigma_w^2\\
&+\frac{LN_{\mathcal{E}}}{2}A^{(t)}+\frac{N_{\mathcal{E}}}{2\eta_w}\mathbb{E}\big[\big\|\boldsymbol{w}^{(t)}-\boldsymbol{w}\big\|^2\big]-\frac{N_{\mathcal{E}}}{2\eta_w}\mathbb{E}\big[\big\|\boldsymbol{w}^{(t+1)}-\boldsymbol{w}\big\|^2\big]. \quad (27)
\end{aligned}
$$

Summing (27) over t and dividing both sides by T, we have

$$
\begin{aligned}
\mathrm{C6}&\leq\frac{N_{\mathcal{E}}\mathbb{E}\big[\big\|w^{(0)}-w\big\|^2\big]}{2\eta_wT}+\frac{LN_{\mathcal{E}}}{2T}\sum_{t=1}^{T}A^{(t)}+\frac{\eta_wN_{\mathcal{E}}G_w^2}{2}+\frac{\eta_w\sigma_w^2}{2N_0}\\
&\overset{(a)}{\leq}\frac{N_{\mathcal{E}}R_{\mathcal{W}}^2}{2\eta_wT}+\frac{\eta_wN_{\mathcal{E}}}{2}G_w^2+10LN_{\mathcal{E}}\eta_w^2\tau_1^2\Big(\frac{m+1}{m}\sigma_w^2+\Psi\Big)\\
&\quad+\frac{\eta_w}{2N_0}\sigma_w^2+10LN_{\mathcal{E}}\eta_w^2\tau_1^2\tau_2^2\Big(\frac{m_{\mathcal{E}}+1}{N_0}\sigma_w^2+\Psi\Big), \quad (28)
\end{aligned}
$$

where (a) is from the bound on $A^{(t)}$ in Lemma 1. Substituting the bound on C5 in (26) and the bound on C6 in (28) into (24), we complete the proof. □