This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

Group Formation and Sampling in Group-based Hierarchical Federated Learning

Jiyao Liu, Xuanzhang Liu, Xinliang Wei, *Member, IEEE*, Hongchang Gao, and Yu Wang, *Fellow, IEEE*

Abstract—Hierarchical federated learning has emerged as a pragmatic approach to addressing scalability, robustness, and privacy concerns within distributed machine learning, particularly in the context of edge computing. This hierarchical method involves grouping clients at the edge, where the constitution of client groups significantly impacts overall learning performance, influenced by both the benefits obtained and costs incurred during group operations (such as group formation and group training). This is especially true for edge and mobile devices, which are more sensitive to computation and communication overheads. The formation of groups is critical for group-based hierarchical federated learning but often neglected by researchers, especially in the realm of edge systems. In this paper, we present a comprehensive exploration of a group-based federated edge learning framework utilizing the hierarchical cloud-edge-client architecture and employing probabilistic group sampling. Our theoretical analysis of its convergence rate, considering the characteristics of client groups, reveals the pivotal role played by group heterogeneity in achieving convergence. Building on this insight, we introduce new methods for group formation and group sampling, aiming to mitigate data heterogeneity within groups and enhance the convergence and overall performance of federated learning. Our proposed methods are validated through extensive experiments, demonstrating their superiority over current algorithms in terms of prediction accuracy and training cost.

Index Terms—Hierarchical federated learning, non-IID, group formation, group sampling, distributed learning, edge computing

1 Introduction

Hierarchical federated learning (HFL) [2], [3] has emerged as a more pragmatic federated learning (FL) [4] paradigm in terms of scalability, efficiency, robustness, and privacy protection. Federated learning, whose fundamental purpose protecting users' data privacy, is actually born hierarchical [3]: clients are typically divided into groups to reduce the communication and computation costs associated with secure aggregation [5]. Given that edge servers can significantly enhance scalability, connection stability, and system robustness, deploying the HFL framework within a client-edge-cloud architecture, as depicted in Fig. 1, is a natural progression.

In this paper, we consider group-based federated edge learning (**Group-FEL**) in such an HFL framework over the client-edge-cloud architecture. First, each edge server manages a set of clients and groups them based on a specific policy. The group information is transmitted to the cloud. The cloud adopts a probabilistic group sampling, selecting a subset of client groups from all groups to perform HFL training at each global round. Clients within a selected group download the global model, train it with their own datasets, and send local updates to the edge server for group aggregation. Group-level operations, such as secure aggregation and backdoor detection, take place during this aggregation. Each group iteratively conducts this in-group workflow before forwarding the aggregated updates to the cloud aggregator at certain rounds. The cloud aggregator performs the final global aggregation and subsequently sends the latest global model back to the edge servers and mobile clients. It's noteworthy that in scenarios where there is only one group on each edge server, the system reverts to a conventional client-edge-cloud HFL configuration.

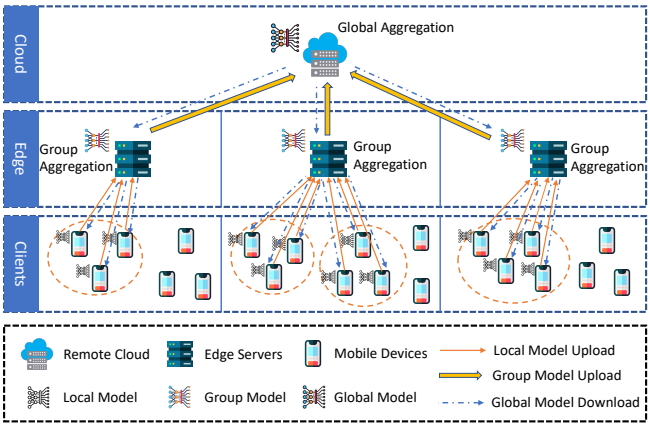


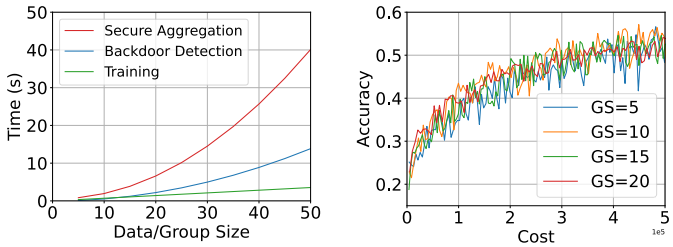
Fig. 1. Group-based federated edge learning (Group-FEL) via the client-edge-cloud architecture. Edge servers perform the group formation and group aggregation, while the remote cloud performs the group sampling and global aggregation. In this example, four client groups are formulated.

Group-FEL gains benefits from the edge network due to the low communication cost, more stable connections, and a large number of connected clients. However, group operations may still incur potentially high costs, which have not yet been well studied by previous works. In Fig. 2(a), we illustrate the overheads incurred at a client through a real-world measurement of a group-based FEL implemented on

• *J. Liu, X. Liu, H. Gao, and Y. Wang are with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19112, USA. {jiyao.liu,xzliu,hongchang.gao,wangyu}@temple.edu. X. Wei is with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518055, China. xl.wei@siat.ac.cn. Wei and Wang are co-corresponding authors.* *This work is partially supported by US NSF under Grant No. CNS-2006604, CNS-2128378 and OAC-2417716. A preliminary version of this paper was appeared in [1].*

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865



(a) group overheads (b) group size (GS)

Fig. 2. Overhead of Group-FEL: (a) different group overheads of a client in a Raspberry Pi edge system: for training cost, x-axis is training data number; for secure aggregation and backdoor detection, x-axis is group size; (b) accuracy over cost with different group sizes.

Raspberry Pi based edge systems. The plot presents three types of overheads: training cost, secure aggregation cost, and backdoor detection cost. Notably, the overheads related to group operations can be comparable to or even significantly surpass the training cost. Considering that clients typically possess a limited amount of data, the overheads associated with group operations will overwhelmingly dominate the costs when the group size is large. This is critical for HFL using mobile or IoT devices as clients, due to their constrained resources.

Simply reducing the group size, however, may not help. As shown in Fig. 2(b), even with a reduction in group size from 20 to 5, the total cost (defined lately as Eq. (5)) required to achieve a certain level of accuracy remains similar. Here, the plotted cost represents the total cost during the entire FL training period including both training and group operation costs. The rationale behind the observation that a smaller group size does not always lead to a reduction in total cost is that data within smaller client groups tends to be more skewed. This skewness hampers the convergence, thereby resulting in higher costs. Numerous FL studies [6]–[14] have highlighted that the non-IID (non-Independent and Identically Distributed) issue significantly hinders the FL convergence.

In this paper, for Group-FEL, we first theoretically derive its convergence rate and show that the data distribution within groups (i.e., the IID degree of the group data) indeed affects the training convergence in theory. Based on such an observation, we introduce a new grouping method (i.e., CoV-Grouping), which leverages the coefficient of variation (CoV) to form client groups. Instead of simply playing trade-offs between the group size and the IID degree (or between cost and accuracy), our group formation method generates smaller groups with less skewed data. In that way, it is possible to form groups that are beneficial to convergence and are less costly. In addition, we further propose different CoV-based sampling methods to calculate group sampling probabilities, so that priority is given to groups with better CoV values. Our results also show that CoV is effective in judging the quality of a group. This observation is also useful for other HFL-based methods. Finally, we compare our proposed method with existing non-IID countering methods via extensive experiments, including the training method based approaches (FedProx [7] and SCAFFOLD [8]), client assignment based approaches (OUEA [15] and SHARE [16]), and a personalized FL approach (FedCLAR [13]). Our results confirm the advances of our proposed method over these existing methods in a group-based HFL setting.

In short, our contributions are summarized as follows:

- We introduce a general group-based hierarchical federated edge learning framework (Group-FEL) where edge servers perform client grouping and the cloud performs probabilistic group sampling.
- We provide a theoretical convergence analysis of Group-FEL with emphasis on the quality of group data distribution, and discovery that group heterogeneity plays an important role in the convergence. This result applies to all generic HFL systems[1].
- We design a new group formation algorithm based on the group's coefficient of variation (CoV), to generate groups with better data distribution, thus speeding up convergence and reducing costs.
- We also propose several group sampling strategies to sample groups with better distribution. The results also shed light on other group sampling methods.
- Extensive experiments are conducted to demonstrate the effectiveness of the proposed group formation algorithm, sampling strategies, as well as the training result of the whole system.

To the best of our knowledge, this is the first work that considers the impact of group overhead in HFL framework over edge systems, and offers a pioneer yet comprehensive exploration, including theoretical analysis, group formation, and sampling strategy specifically designed for Group-FEL. We hope this works inspires more future investigations into this important but ignored problem. The remaining of this paper is organized as follows. Section 2 introduces the Group-FEL framework, while its detailed convergence analysis are provided in Section 3. Based on the observation from the analysis, we then propose our group formation algorithm in Section 4 and group sampling methods in Section 5. Evaluations of the proposed methods are provided in Section 6. Section 7 reviews related works. Section 8 concludes the paper with possible future directions.

2 Group-based HFL Framework

We now introduce the system architecture, learning algorithm, and cost model in the studied Group-FEL framework.

2.1 System Architecture

In this paper, we consider a group-based hierarchical federated learning over edge computing, as shown in Fig. 1. We assume that multiple mobile clients (let $\mathcal{C}$ be the client set) are connected to cloud via edge servers. Each edge server will divide its clients ($\mathcal{C}_j$, the client set of j-th edge server) into multiple mutually exclusive client groups. Let $\mathcal{G}$ and $\mathcal{G}_j$ be the set of all groups and the set of groups of j-th edge server, respectively. Then the federated learning is performed with selected client groups (denoted by $\mathcal{S}_t$) based on a certain selection mechanism (i.e., via group sampling with a probability vector $\mathbf{p}$) at each global round

1. Our theoretical analysis does not rely on a specific grouping method, thus is general enough to cover any generic group-based HFL. Furthermore, since we consider both the quality of the group data distribution and the probability of sampling the group, our result is different from the existing HFL convergence analysis (e.g., [2] and [15]).

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

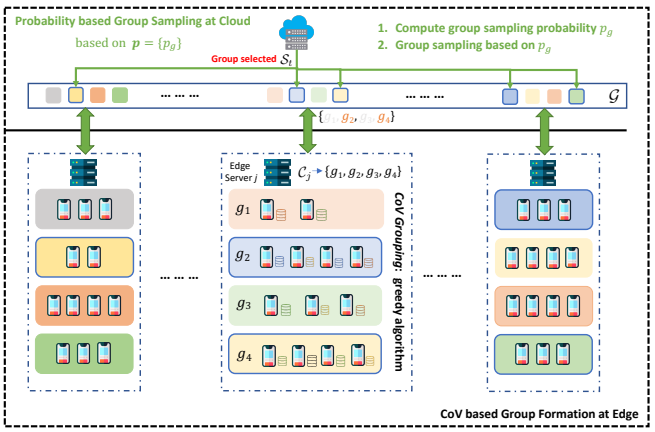


Fig. 3. The overall grouping framework in Group-FEL: (lower) *CoV-based group formation* at the edge, in this example, four groups are formed at each edge server; and (upper) *probability based group sampling* at the cloud, where the five selected groups are framed.

t. Each client c_i in a selected group performs local training and sends its local model updates to its edge server for group aggregation. Then edge servers will perform group aggregation and submit the group model updates to the cloud server (i.e., parameter server) for global aggregation. Fig. 3 illustrates the overall grouping framework of Group-FEL, where the grouping is performed at the edge server for its clients, and the group sampling is done at the cloud with the sampling probability vector.

2.2 Learning Algorithm

The overall training algorithm of group-based FEL is shown in Algorithm 1. In Algorithm 1, Lines 2-3 are for group formation at each edge server, and Line 4 is for the computation of sampling probability vector $\mathbf{p}$ of all groups at the cloud. These are important steps for Group-FEL, thus we will present our detailed design of them in Sections 4 and 5, respectively. Lines 5-15 are the group-based federated learning steps, which include group sampling (Line 6), local update (Line 13), group aggregation (Line 14), and global aggregation (Line 15). Here, x_t, $x^g_{t,k}$, $x^i_{t,k,e}$ represent the global model at t-th global round, the group model at k-th group round within t-th global round, the local model of client c_i at e-th local round within k-th group round and t-th global round, respectively.

In classic federated learning, given the client set $\mathcal{C}$ with N clients, and the loss function f_i of the client c_i, we have the global loss function

$$f(x) = \sum_{c_i \in \mathcal{C}} \frac{n_i}{n} f_i(x), \tag{1}$$

where n_i is the data entry number on i-th client, and $n = \sum_{i=0}^{N-1} n_i$.

When we divide clients into a set of groups $\mathcal{G}$, then for each group $\mathbf{g}$, its loss function is

$$f_g(x) = \sum_{c_i \in \mathbf{g}} \frac{n_i}{n_g} f_i(x), \tag{2}$$

where n_g is the number of data on all clients inside the group $\mathbf{g}$. Hence, the global loss function can be rewritten as

$$f(x) = \sum_{\mathbf{g} \in \mathcal{G}} \frac{n_g}{n} f_g(x). \tag{3}$$

Algorithm 1 GROUP-BASED FEDERATED EDGE LEARNING

Input: Client sets $\mathcal{C}_j$, number of sampled groups in each round $S = |\mathcal{S}_t|$, initial global model x_0, global round T, group round K, local round E, learning rate η.
Output: Final global model x_{T-1}.

1: $\mathcal{G} = \emptyset$
2: **for** each client set $\mathcal{C}_j$ **do** ▷ *in parallel*
3: $\quad \mathcal{G} = \mathcal{G} \cup$ CoV-GROUPING$(\mathcal{C}_j)$ ▷ *group formation*
4: $\mathbf{p} =$SAMPLING-PROB$(\mathcal{G})$ ▷ *group sampling prob*
5: **for** t from 0 to $T-1$ **do**
6: $\quad$ Sample $\mathcal{S}_t \subseteq \mathcal{G}$ according to $\mathbf{p}$ ▷ *group sampling*
7: $\quad$ **for** group $\mathbf{g}$ in $\mathcal{S}_t$ **do** ▷ *in parallel*
8: $\quad\quad x^g_{t,0} = x_t$ ▷ *initialize group model*
9: $\quad\quad$ **for** k from 0 to $K-1$ **do**
10: $\quad\quad\quad$ **for** client c_i in group g **do** ▷ *in parallel*
11: $\quad\quad\quad\quad x^i_{t,k,0} = x^g_{t,k}$ ▷ *initialize client model*
12: $\quad\quad\quad\quad$ **for** e from 0 to $E-1$ **do**
13: $\quad\quad\quad\quad\quad x^i_{t,k,e+1} = x^i_{t,k,e} - \eta \nabla f_i(x^i_{t,k,e}; \xi^i_{t,k,e})$ ▷ *local update*
14: $\quad\quad\quad x^g_{t,k+1} = \sum_{i \in g} \frac{n_i}{n_g} x^i_{t,k,E-1}$ ▷ *group aggregation*
15: $\quad x_{t+1} = \sum_{g \in \mathcal{S}_t} \frac{n_g}{n_t} x^g_{t,K-1}$ ▷ *global aggregation*
16: **return** x_{T-1}

At Line 15 of Algorithm 1, the global aggregation may lead to the learned model biased since some groups have higher probability to be sampled during the group sampling. This is true for our design since we always give higher priority to groups with better distribution to boost convergence. Therefore, we will discuss this in Section 5. If the model is required to be unbiased, a correction factor $\frac{1}{p_g S}$ can be introduced and Line 15 is then replaced by

$$x_{t+1} = \sum_{g \in \mathcal{S}_t} \frac{1}{p_g S} \cdot \frac{n_g}{n} x^g_{t,K-1}, \tag{4}$$

where p_g is the probability to sample the group $\mathbf{g}$ during the group sampling and S is the number of sampled groups in each round $S = |\mathcal{S}_t|$. Note that n_t in Line 15 of Algorithm 1 in the number of data entries on the t-th global round.

2.3 Cost Model

To measure the learning cost of Group-FEL, we focus on both computation and communication loads on all clients. Each client has two types of costs: *training cost* and *group operation cost*.

The built-in training cost $\mathcal{H}_i(n_i)$ of client c_i measures the time needed to iterate through its trainset once. Given the hardware, model, and training hyperparameters are fixed, this cost is proportional to the data sample number n_i owned by this client.

Overheads incurred by group operations (both for secure or privacy-preserving computation and communication) are quadratic to the group size $|\mathbf{g}|$ [5], [17]. These two assumptions can be further confirmed by our experiments shown in Fig. 9. Hereafter, we use $\mathcal{O}_g(|\mathbf{g}|)$ to denote the group overhead of each client in group $\mathbf{g}$. This group overhead cost is often ignored in the analysis of existing works.

By adding training costs and group operation costs of all clients in each group, the total learning costs in the whole training process can be measured by

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

$$\mathcal{O} = \sum_{t=0}^{T-1} \left(\sum_{\mathbf{g} \in \mathcal{S}_t} K \sum_{c_i \in \mathbf{g}} (\mathcal{O}_g(|\mathbf{g}|) + E\mathcal{H}_i(n_i)) \right). \quad (5)$$

In our evaluations, we will measure the performance of all algorithms using the achieved accuracy by certain learning costs instead of the accuracy by the global round.

3 Convergence Analysis

We now present our main theorem on the convergence of Group-FEL with an emphasis on the group characters. This result is an important theoretical contribution and also inspires the design of our grouping formation and sampling schemes. The result applies to all HFL structures where an intermediary aggregation layer is used. The result is also general enough to cover existing convergence results[2].

3.1 Assumptions and Lemmas

To perform convergence analysis, previous works [18]–[20] have made common assumptions about the local and global loss functions. We simply borrow and list them here.

Assumption 1. **Bounded variance of local gradient:** for any local loss function f_i,

$$\|\nabla f_i(x; \xi^i) - \nabla f_i(x)\|^2 \leq \sigma^2. \quad (6)$$

Here ξ^i denotes the data used to compute the gradient at client c_i in a certain round. Clients may not always use all data to calculate the gradient, thus there is a variance compared to the full gradient.

Assumption 2. **L-smoothness:** for any local loss function f_i (also global loss function f),

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|. \quad (7)$$

Assumption 3. **Bounded local heterogeneity:** for any local loss function f_i,

$$\|\nabla f_i(x) - \nabla f(x)\|^2 \leq \zeta^2. \quad (8)$$

Assumption 4. **Bounded group heterogeneity:** the heterogeneity between any group loss function f_g and global loss function f satisfies

$$\|\nabla f_g(x) - \nabla f(x)\|^2 \leq \zeta_g^2. \quad (9)$$

Here ζ_g is a constant that measures the heterogeneity between any f_g and f. There is no practical way to compute ζ_g and L but it is generally believed that ζ_g relies on the difference between the global and group data distributions, i.e., the more similar the two distributions are, the smaller ζ_g is. Note that although ζ_g and ζ are both on heterogeneity they are quite different in our design. ζ reflects the heterogeneity caused by individual clients which cannot be controlled, while ζ_g is the heterogeneity of the formed groups. In Group-FEL, if we can **control the group formation smartly to reduce** ζ_g, then the selected groups will be more IID, thus leading to better performance.

In the proofs of two lemmas needed for our main theorem, we will use the following lemmas on the properties of f_g with local update (i.e., $F_g = \sum_{i \in g} \frac{n_i}{n_g} \sum_{e=0}^{E-1} \nabla f_i(x_e^i; \xi_e^i)$) and the bound of $\sum_{g \in \mathcal{G}} \|x_{t,k}^g - x_t\|^2$. The proof of these lemmas are provided as supplemental material.

2. When $|\mathcal{S}_t| = |\mathcal{G}|$, it degrades to HFL without group sampling. When there is only one group on each edge server, it degrades to the classic HFL.

Lemma 1. The gradient variance of F_g is bounded by

$$\|\nabla F_g(x; \xi_g) - \nabla F_g(x)\|^2 \leq \gamma(1 + 10\eta^2 E^2 L^2) E^2 \sigma^2.$$

Lemma 2. The smoothness of F_g is bounded by

$$\|\nabla F_g(x) - \nabla F_g(y)\|^2 \leq 6\gamma E^2 L^2 \|x - y\|^2 + 10\gamma\eta^2 E^4 L^2 \sigma^2.$$

Lemma 3. The heterogeneity of F_g is bounded by

$$\|\nabla F_g(x) - E\nabla f(x)\|^2 \leq (\frac{3\gamma E}{|\mathbf{g}|} + 15\gamma\eta^2 E^3)\sigma^2 + 90\gamma\eta^2 E^4 \zeta^2 + 3E^2\zeta_g^2 + 90\gamma\eta^2 E^2 L^2 \|E\nabla f(x)\|^2.$$

Lemma 4. $\sum_{g \in \mathcal{G}} \|x_{t,k}^g - x_t\|^2$ is bounded by

$$\frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \|x_{t,k}^g - x_t\|^2 \leq \lambda_\sigma \gamma \sigma^2 + \lambda_3 \zeta^2 + 90\eta^2 K^2 E^2 \zeta_g^2 + \lambda_f \mathbb{E}[\|\eta E \nabla f(x_t)\|^2].$$

Here, the constants $\lambda_\sigma, \lambda_3, \lambda_f$ are defined in Theorem 5.

3.2 Main Theorem on Convergence

Theorem 5. The convergence rate of Group-FEL is bounded as follows,

$$\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq \frac{f(x_0) - \mathbb{E}[f(x_T)]}{\lambda_1 \eta T K E} + \frac{\lambda_s \cdot \frac{\Gamma_p}{|\mathcal{S}_t|}}{\lambda_1 K E} + \frac{\gamma\Gamma(\lambda_2\sigma^2 + \lambda_3\zeta^2 + \lambda_4\zeta_g^2)}{\lambda_1}. \quad (10)$$

Here η is the learning rate, and γ, Γ, Γ_p and constants $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_s, \lambda_\sigma, \lambda_f$ are defined or constrained by following (to simplify the expression of the final result in the proofs)

$$\gamma = |\mathbf{g}|^2 [\frac{1}{|\mathbf{g}|^2} + Var(\frac{n_i}{n_g})], \quad (11)$$

$$\Gamma = |\mathcal{G}|^2 [\frac{1}{|\mathcal{G}|^2} + Var(\frac{n_g}{n})], \ \ \Gamma_p \geq \sum_{\mathbf{g} \in \mathcal{G}} \frac{1}{p_g}, \quad (12)$$

$$\lambda_s = \eta\gamma\Gamma K^2 (1 + 10\eta^2 E^2 L^2 \sigma^2), \quad (13)$$

$$0 < \lambda_1 \leq \frac{1}{2} - 3\lambda_f \eta\gamma\Gamma K E L^2, \quad (14)$$

$$\lambda_2 = 3\lambda_\sigma \gamma L^2 + 5\eta^2 E^2 L^2, \quad (15)$$

$$\lambda_3 = 2700\eta^4 \gamma K^2 E^4 L^2, \ \ \lambda_4 = 90\eta^2 K^2 E^2 L^2, \quad (16)$$

$$\lambda_f = 30\eta^2 K^2 (1 + 90\gamma\eta^2 E^2 L^2), \quad (17)$$

$$\lambda_\sigma = 5K\eta^2 E^2 [1 + ((1 + 6K)E + 9K)10\eta^2 E L^2 + \frac{18K}{|\mathbf{g}|E}], \quad (18)$$

$$\eta \leq \frac{1}{2KE}, \quad \eta \leq \frac{1}{\sqrt{6(E-1)(2E-1)L^2}}, \quad (19)$$

$$\eta \leq \frac{1}{\sqrt{36\gamma(K-1)(2K-1)E^2L^2}}). \quad (20)$$

Proof: Due to the space limit, we cloud not include all proof details. Here we only present a brief proof skeleton. First, we consider another form of Assumption 2, as

$$f(y) \leq f(x) + \langle \nabla f(x), (y - x) \rangle + \frac{L}{2}\|x - y\|^2. \quad (21)$$

Here, $\langle , \rangle$ is the vector inner production operator. Based on this smoothness assumption, we can have

$$\mathbb{E}\left[f(x_{t+1})\right] \leq f(x_t) - \eta K E \|\nabla f(x_t)\|^2 + \frac{L}{2}\mathbb{E}_t\|\Delta_t\|^2 + \langle \nabla f(x_t), \mathbb{E}_t\left[\Delta_t + \eta K E \nabla f(x_t)\right] \rangle, \quad (22)$$

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

where $\Delta_t = x_{t+1} - x_t = \sum_{\mathbf{g}\in\mathcal{S}_t} \frac{n_g}{n_t} \sum_{k=0}^{K-1} \eta \nabla F_g(x_{t,k})$ is the global update at the round t and $F_g(x) = \sum_{i\in\mathbf{g}} \frac{n_i}{n_g} \sum_{e=0}^{E-1} f_i(x_e)$ is single round group update. By defining $A_1 = \langle \nabla f(x_t), \mathbb{E}_t [\Delta_t + \eta KE \nabla f(x_t)] \rangle$ and $A_2 = \mathbb{E}_t \|\Delta_t\|^2$, we can rewrite Eq. (22) as

$$\mathbb{E}[f(x_{t+1})] \leq f(x_t) - \eta KE \|\nabla f(x_t)\|^2 + A_1 + \frac{L}{2} A_2. \quad (23)$$

Then we bound A_1 and A_2 by proving Lemma 6 and Lemma 7, respectively. Proofs of these two lemmas are provided in the next subsection. Bringing such bounds into Eq. (23), we then have

$$\begin{aligned}\mathbb{E}[f(x_{t+1})] \leq & f(x_t) - \eta KE (\frac{1}{2} - 3\lambda_f \eta\gamma\Gamma KEL^2) \|\nabla f(x_t)\|^2 \\ & + \lambda_2 \eta KE\gamma\Gamma\sigma^2 + \lambda_3 \eta KE\gamma\Gamma\zeta^2 + \lambda_4 \eta KE\gamma\Gamma\zeta_g^2 \\ & + \frac{\eta^2\gamma\Gamma\Gamma_p K^2}{|\mathcal{S}_t|}(1 + 10\eta^2 E^2 L^2 \sigma^2) \\ & + (\eta^2 - \frac{n}{2KE}) \mathbb{E}_t \left[\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2 \right]. \end{aligned} \quad (24)$$

With conditions of Eqs. (13)-(20), we can show

$$\begin{aligned}\mathbb{E}[f(x_{t+1})] \leq & f(x_t) - \lambda_1 \eta KE \|\nabla f(x_t)\|^2 + \eta\gamma\Gamma KE \\ & (\lambda_2\sigma^2 + \lambda_3\zeta^2 + \lambda_4\zeta_g^2) + \eta \cdot \lambda_s \cdot \frac{\Gamma_p}{|\mathcal{S}_t|}. \end{aligned} \quad (25)$$

Finally, by rearranging and telescoping, we have

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq & \frac{f(x_0) - \mathbb{E}[f(x_T)]}{\lambda_1 \eta TKE} + \frac{\lambda_s \cdot \frac{\Gamma_p}{|\mathcal{S}_t|}}{\lambda_1 KE} \\ & + \frac{\gamma\Gamma(\lambda_2\sigma^2 + \lambda_3\zeta^2 + \lambda_4\zeta_g^2)}{\lambda_1}. \end{aligned}$$

□

Recall that T, K, E, and η are the number of global rounds, the number of group rounds in each global round, the number of local rounds in each group round, and the learning rate in local updating, respectively, as defined in Algorithm 1. The inequality in Theorem 5 (i.e., Eq. (10)) tells us that when the right-hand side is diminishing as T increases (if properly select the learning rate η, e.g., $\eta = \frac{1}{\sqrt{T}}$), the gradient norm $\|\nabla f(x_t)\|$ tends to be zero, which means the model converges to a local minimum.

3.3 Proofs of Lemmas 6 and 7

We now prove Lemmas 6 and 7 on bounds A_1 and A_2, which are used in the above proof of the main theorem.

Lemma 6. Under the assumptions and conditions in Theorem 5, A_1 is bounded as follows

$$\begin{aligned}A_1 \leq & \eta KE (\frac{1}{2} + 90\eta^3\gamma\Gamma K^3 E^3 L^2) \|\nabla f(x_t)\|^2 \\ & + \lambda_2 \eta KE\gamma\Gamma\sigma^2 + \lambda_3 \eta KE\gamma\Gamma\zeta^2 + \lambda_4 \eta KE\gamma\Gamma\zeta_g^2 \\ & - \frac{\eta}{2KE} \mathbb{E}_t \left[\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2 \right]. \end{aligned} \quad (26)$$

Proof: Since $A_1 = \langle \nabla f(x_t), \mathbb{E}_t [\Delta_t + \eta KE \nabla f(x_t)] \rangle$,

$$\begin{aligned}A_1 = & \langle \nabla f(x_t), \mathbb{E}_t [(\sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \sum_{i\in g} \frac{n_i}{n_g} \sum_{e=0}^{E-1} -\eta \nabla f_i(x_{t,k,e}; \xi_{t,k,e}^i)) \\ & + \eta KE \nabla f(x_t)] \rangle \\ = & \langle \sqrt{\eta KE} \nabla f(x_t), -\frac{\sqrt{\eta}}{\sqrt{KE}} \mathbb{E}_t [(\sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)) \\ & - KE \nabla f(x_t)] \rangle \\ \leq & \frac{\eta}{2KE} \mathbb{E}_t [\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} (\nabla F_g(x_{t,k}^g) - \nabla F_g(x_t)) \|^2] \\ & + \frac{\eta KE}{2} \|\nabla f(x_t)\|^2 - \frac{\eta}{2KE} \mathbb{E}_t [\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2]. \end{aligned}$$

Due to the smoothness of of F_g (Lemma 2), we have

$$\begin{aligned}A_1 \leq & \frac{\eta|\mathcal{G}|}{2E} \mathbb{E}_t \left[\sum_{g\in\mathcal{G}} \frac{n_g^2}{n^2} \sum_{k=0}^{K-1} \left(6\gamma E^2 L^2 \|x_{t,k}^g - x_t\|^2 + 10\gamma\eta^2 E^4 L^2 \sigma^2 \right) \right] \\ & + \frac{\eta KE}{2} \|\nabla f(x_t)\|^2 - \frac{\eta}{2KE} \mathbb{E}_t \left[\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2 \right]. \end{aligned} \quad (27)$$

Based on Lemma 4 which bounds $\|x_{t,k}^g - x_t\|^2$, we have

$$\begin{aligned}A_1 \leq & 3\eta\gamma\Gamma KEL^2 \mathbb{E}_t [\lambda_\sigma \gamma\sigma^2 + 900\eta^4 K^2 E^4 L^2 \gamma\zeta^2 + 30\eta^2 K^2 E^2 \zeta_g^2 \\ & + \lambda_f \mathbb{E}[\|\eta E \nabla f(x_t)\|^2]] + 5\eta^3\gamma\Gamma KE^3 L^2 \sigma^2 + \frac{\eta KE}{2} \|\nabla f(x_t)\|^2 \\ & - \frac{\eta}{2KE} \mathbb{E}_t \left[\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2 \right] \\ \leq & \eta KE (\frac{1}{2} + 3\lambda_f \eta\gamma\Gamma KEL^2) \|\nabla f(x_t)\|^2 + \lambda_2 \eta KE\gamma\Gamma\sigma^2 \\ & + \lambda_3 \eta KE\gamma\Gamma\zeta^2 + \lambda_4 \eta KE\gamma\Gamma\zeta_g^2 \\ & - \frac{\eta}{2KE} \mathbb{E}_t \left[\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2 \right]. \end{aligned}$$

□

Lemma 7. Under the assumptions and conditions in Theorem 5, A_2 is bounded as follows

$$\begin{aligned}A_2 \leq & \eta^2 \mathbb{E}_t \left[\| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g) \|^2 \right] \\ & + \frac{\eta^2\gamma\Gamma\Gamma_p K^2}{|\mathcal{S}_t|}(1 + 10\eta^2 E^2 L^2 \sigma^2). \end{aligned} \quad (28)$$

Proof: Recall that $A_2 = \mathbb{E}_t \|\Delta_t\|^2$. Thus,

$$\begin{aligned}A_2 = & \mathbb{E}_t \| \sum_{g\in\mathcal{S}_t} \frac{1}{p_g |\mathcal{S}_t|} \cdot \frac{n_g}{n} \Delta_t^g \|^2 \\ = & \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \mathbb{I}\{g \in \mathcal{S}_t\} \frac{1}{p_g |\mathcal{S}_t|} \cdot \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g; \xi_{t,k}^g) \|^2. \end{aligned}$$

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

Since $E[X^2] = E[X - E[X]]^2 + E[X]^2$, we know

$$\begin{aligned}
A_2 &= \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{\mathbb{P}\{g \in \mathcal{S}_t\}}{p_g|\mathcal{S}_t|} \cdot \frac{n_g}{n} \sum_{k=0}^{K-1} (\nabla F_g(x_{t,k}^g; \xi_{t,k}^g) \\
&\quad - \nabla F_g(x_{t,k}^g))\|^2 + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2 \\
&= \eta^2 |\mathcal{G}| K \mathbb{E}_t \sum_{g\in\mathcal{G}} \frac{\mathbb{P}\{g \in \mathcal{S}_t\}}{p_g^2|\mathcal{S}_t|^2} \cdot \frac{n_g^2}{n^2} \sum_{k=0}^{K-1} \|\nabla F_g(x_{t,k}^g; \xi_{t,k}^g) \\
&\quad - \nabla F_g(x_{t,k}^g)\|^2 + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2 \\
&= \eta^2 |\mathcal{G}| K \mathbb{E}_t \sum_{g\in\mathcal{G}} \frac{1}{p_g|\mathcal{S}_t|} \cdot \frac{n_g^2}{n^2} \sum_{k=0}^{K-1} \|\nabla F_g(x_{t,k}^g; \xi_{t,k}^g) \\
&\quad - \nabla F_g(x_{t,k}^g)\|^2 + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2 \\
&= \frac{\eta^2 |\mathcal{G}| K}{|\mathcal{S}_t|} \mathbb{E}_t \sum_{\mathbf{g}\in\mathcal{G}} \frac{1}{p_g} \sum_{\mathbf{g}\in\mathcal{G}} \frac{n_g^2}{n^2} \sum_{g\in\mathcal{G}} \sum_{k=0}^{K-1} \|\nabla F_g(x_{t,k}^g; \xi_{t,k}^g) \\
&\quad - \nabla F_g(x_{t,k}^g)\|^2 + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2.
\end{aligned} \tag{29}$$

Due to the definition of Γ and Γ_p, we then have

$$\begin{aligned}
A_2 &\le \frac{\eta^2 |\mathcal{G}| K}{|\mathcal{S}_t|} \Gamma_p \frac{\Gamma}{|\mathcal{G}|^2} \mathbb{E}_t \sum_{g\in\mathcal{G}} \sum_{k=0}^{K-1} \|\nabla F_g(x_{t,k}^g; \xi_{t,k}^g) - \nabla F_g(x_{t,k}^g)\|^2 \\
&\quad + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2 \\
&\le \frac{\eta^2 K^2 \Gamma_p \Gamma}{|\mathcal{S}_t|} \mathbb{E}_t \|\nabla F_g(x_{t,k}^g; \xi_{t,k}^g) - \nabla F_g(x_{t,k}^g)\|^2 \\
&\quad + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2.
\end{aligned}$$

Due to the definition of γ and Lemma 1 on the bounded variance of gradient of F_g, we have

$$\begin{aligned}
A_2 &\le \frac{\eta^2 \gamma \Gamma \Gamma_p K^2 E^2}{|\mathcal{S}_t|} (1 + 10\eta^2 E^2 L^2)\sigma^2 \\
&\quad + \eta^2 \mathbb{E}_t \| \sum_{g\in\mathcal{G}} \frac{n_g}{n} \sum_{k=0}^{K-1} \nabla F_g(x_{t,k}^g)\|^2.
\end{aligned}$$

□

3.4 Key Observations

From the main theorem, we can have the following observations, which inspire our proposed group formation and sampling methods.

First, the heterogeneity ζ_g between the combined group loss function f_g and the global loss functions f plays a role in convergence. With a larger heterogeneity, the convergence will be slower. Therefore, in our proposed group formation and group sampling schemes, we aim to reduce this heterogeneity. Unfortunately, the definition of ζ_g is not straightforward and we cannot directly compute it to quantify the heterogeneity. Therefore, instead, we use the difference between data distributions to measure how analogous two loss functions are. Concretely, we use the *Coefficient of Variance* (CoV) of the labels in a group, which will be discussed in the next section. The CoV-based grouping algorithm bridges this observation to our system design.

Second, the larger variance of sampling vector $\mathbf{p}$ (thus larger Γ_p) may also delay the convergence of unbiased sampling. Due to the unbiasedness factor $\frac{1}{p_g|\mathcal{S}_t|}$, any $\frac{1}{p_g}$ should not be too large, otherwise the aggregation is numerically unstable: $\frac{1}{p_g}$ extremely amplifies the gradient and ruins all previous training results. On the other hand, we want to be able to set arbitrary $\mathbf{p}$ to prioritize good groups with smaller ζ_g. To handle this, when we adopt unbiased aggregation together with prioritized sampling, we will use a normalization method (see Section 5). Note that $\mathbf{p}$ may also entangle with ζ_g, γ, and other group-specific characters as it decides which groups participate in the training more, and therefore their characters affect the FL system more.

Third, to boost convergence, we need a smaller γ. We find that $\gamma - 1 = |\mathbf{g}|^2 Var(\frac{n_i}{n_g}) = (\frac{\sigma_c}{\mu_c})^2$, where σ_c and μ_c are the standard deviation and mean of the total data sample number among clients within the same group. Interestingly, $\gamma - 1$ is also the square of CoV of data sample number within the group. Γ has a similar property. Reducing γ also helps to converge faster and smoother. We leave further considering this in our design as one of the future works.

Note that our analysis hints that methods taking group heterogeneity (ζ_g) into account may perform well in term of convergence. This has been confirmed by our experiment results in Section 6, where our proposed method outperforms several existing methods. FedProx [7] simply shrink the update vector (gradient) to keep the model not too far from the last round's, which is ignorant to group heterogeneity. SCAFFOLD [8] tries to correct the update direction but it may not even know what the good direction is without a good group due to large ζ_g. OUEA [15], SHARE [16] and Fed-CBS [21] try to form good groups (to reduce ζ_g) but the resulting groups are not as good as ours in the HFL setting.

4 CoV based Group Formation

In Group-FEL, how to perform group formation is critical since ζ_g plays an important role in convergence (as suggested by the first key observation above). Again, note that ζ_g depends on the difference between the in-group data distributions and the global data distribution, and a good group formation method helps to reduce ζ_g. Therefore, in this section, we first discuss the possible grouping criteria and then present our proposed grouping method.

4.1 Grouping Criteria: CoV

Based on the key observation from our convergence analysis, the principle of grouping criteria should be to make the group loss functions as similar to the global loss function as possible (i.e., similar data distribution and smaller ζ_g). In general, the properties of a loss function are closely related to its data distributions. Therefore, our grouping criteria aims to measure the similarity of data distributions between group and global. We further assume global data are evenly distributed, thus we can just focus on the data distribution within each local group. To do so, we introduce the *coefficient of variation* (**CoV**) of the labels in a group.

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

Here we focus on the grouping of a client set $\mathcal{K}$, whose i-th client is c_i. The data label set $\mathcal{Y}$ contains m kinds of labels. We define a label matrix $\mathcal{L}$, where $\mathcal{L}_{i,j}$ is the number of j-th category of data samples on i-th client. Then, a grouping $\mathcal{G}$ of $\mathcal{K}$ is a partition of $\mathcal{K}$. Let $\mathcal{G}_l$ be the l-th group in $\mathcal{G}$, which contains all clients in this group. To compute the CoV of a group, we only need to know the data label distributions from users in that group, without any information of their local data, model, nor gradient.

Ideally, we would like the distribution of every group $\mathcal{G}_l$ is identical to the global distribution, i.e.,

$$\frac{\sum_{c_i\in\mathcal{G}_l}\mathcal{L}_{i,j}}{\sum_{c_i\in\mathcal{G}_l}\sum_{k\in\mathcal{Y}}\mathcal{L}_{i,k}}=\frac{\sum_{c_i\in\mathcal{K}}\mathcal{L}_{i,j}}{\sum_{c_i\in\mathcal{K}}\sum_{k\in\mathcal{Y}}\mathcal{L}_{i,k}},\quad\forall j,\forall l. \tag{30}$$

However, such restricted criteria might lead to infeasible grouping. Thus, instead, we use the coefficient of variation as the grouping criterion. For a given group $\mathbf{g}$, we calculate its coefficient of variation (CoV) in the following way

$$CoV(\mathbf{g})=\frac{\sigma(\mathbf{g})}{\mu(\mathbf{g})}=\frac{\sqrt{\sum_{j\in\mathcal{Y}}\left(\frac{n_g}{m}-\sum_{c_i\in\mathbf{g}}\mathcal{L}_{i,j}\right)^2}}{n_g}. \tag{31}$$

Note that the group variance can also be defined as

$$\sigma(\mathbf{g})=\frac{\sqrt{\sum_{j\in\mathcal{Y}}\left(\frac{n_g}{m}-\sum_{c_i\in\mathbf{g}}\mathcal{L}_{i,j}\right)^2}}{m}. \tag{32}$$

Recall that n_g is the number of data samples in the group and m is the number of labels (data samples types). The reason why the variance (i.e., $\sigma^2(\mathbf{g})$) is not suitable as the criterion is that it is susceptible to the scale of data number. For example, a group with a smaller total data number but larger data distribution skew may have a smaller variance than a group with more data but smaller distribution skew. We may prefer the latter group but, on the contrary, the smaller variance criterion prefers the first one. Note that neither variance nor CoV has been considered in previous works on client grouping at edge-based federated learning.

4.2 Group Formation Problem

The group formation problem aims to divide all clients associated with an edge server into multiple client groups such that the summation of CoVs of all groups is minimized. We can use matrices to more succinctly express this grouping problem. Suppose we have three matrices A, X, and B, where A_{ji} is the number of label type j that client c_i possesses; X_{il} is the grouping decision indicator where $X_{il}=1$ if the client c_i is in the group $\mathcal{G}_l$, otherwise 0; B_{jl} is the number of data type j in group l. Then the group formation problem can be formulated as the following optimization problem:

$$\min_X \sum_l \frac{\sqrt{\sum_j\left(\frac{\sum_j B_{jl}}{m}-B_{jl}\right)^2}}{\sum_j B_{jl}} \tag{33}$$

$$\textbf{s.t.}\quad AX=B, \tag{34}$$

$$\sum_i X_{il}\geq MinGS,\quad\forall l, \tag{35}$$

$$\sum_l X_{il}=1,\quad\forall i, \tag{36}$$

$$X_{il}\in\{0,1\},\quad\forall i,\forall l. \tag{37}$$

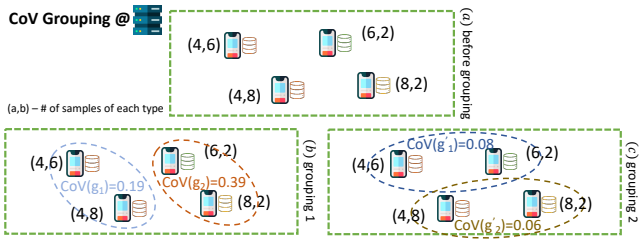


Fig. 4. Example of CoV-Grouping at edge: (a) before grouping, (b) two groups formed by grouping method 1, (c) two groups formed by grouping method 2.

Constraint (35) is an anonymity constraint to make sure that each group at least has $MinGS$ clients[3], while Constraint (36) ensures that a client will be grouped into one and only one group.

Fig. 4 illustrates a simple toy example. As shown in Fig. 4(a), this edge server has four clients and all wants to form a group with at least two clients. Each client has certain amount of data samples of two label types. Though two different grouping methods (as shown by Fig. 4(b) and (c)) can both formulate two groups each with two clients, the total CoVs are quite different. In our design, we would prefer the one with lower CoVs (Fig. 4(c)).

4.3 Grouping Algorithm: CoV Grouping

Directly solving the above grouping optimization is not easy. Note that the grouping problem with a fixed number of groups is a variation of the k-mean clustering problem, which is known as NP-hard [22], [23]. Therefore, we design a greedy algorithm (CoV-Grouping) to generate an approximating solution. It generates groups one by one until no more groups are possible. For each group, it first randomly picks a client, then greedily adds clients one by one. When adding a new client to the current group, it tries every possible client and adds the one that reduces the group CoV the most (Line 5). If no one meets this criterion and the group size is large enough (reach $MinGS$), then this group is finalized and the algorithm starts the next group. In addition, besides checking the group size constraint $MinGS$, we also add a maximum CoV requirement ($MaxCoV$) which makes sure the resulting group CoV is smaller than $MaxCoV$[4]. Algorithm 2 shows the details.

Theorem 8. The time complexity of Algorithm 2 is $O(|\mathcal{K}|^3|\mathcal{Y}|)$, where $|\mathcal{K}|$ and $|\mathcal{Y}|$ are the number of clients and the number of label types, respectively.

Proof: First, Line 5 is executed at most $O(|\mathcal{K}|)$ times. Second, in each execution, it implicitly calls $CoV()$ at most $O(|\mathcal{K}|)$ times to try every possible client. Last, the complexity of function $CoV()$ is $O(|\mathbf{g}||\mathcal{Y}|)=O(|\mathcal{K}||\mathcal{Y}|)$. Therefore, the total time complexity of CoV-Grouping is $O(|\mathcal{K}|^3|\mathcal{Y}|)$. □

3. The minimum group size $MinGS$ can make sure that the secure group operation can protect the model/data privacy of its clients. Here we assume the requirement is a controllable constant for our system, but this can be easily extended to the case where each client has its own group size requirement.

4. This is not a hard constraint, i.e., the algorithm only tries to adding clients until the CoV is satisfied, but sometimes it might be infeasible to reach lower than $MaxCoV$, then it just gives up adding more clients and finalize this group (Line 9 of Algorithm 2).

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

Algorithm 2 CoV-Grouping

Input: Client set $\mathcal{K}$, min GS $MinGS$, and $MaxCoV$.
Output: Group set $\mathcal{G}$.

$\mathcal{G} = \emptyset$
while $\mathcal{K} \neq \emptyset$ **do**
  Find a random $c \in \mathcal{K}$, $\mathbf{g} = \{c\}$, $\mathcal{K} = \mathcal{K} \backslash c$. ▷ *create a new group*
  while (CoV($\mathbf{g}$) $>$ $MaxCoV$ or $|\mathbf{g}| < MinGS$) and $\mathcal{K} \neq \emptyset$ **do** ▷ *not meet the group requirement yet*
    Find $c \in \mathcal{K}$ that minimizes CoV($\mathbf{g} \cup c$)
    if CoV($\mathbf{g} \cup c$) $<$ CoV($\mathbf{g}$) or $|\mathbf{g}| < MinGS$ **then**
      $\mathbf{g} = \mathbf{g} \cup c$, $\mathcal{K} = \mathcal{K} \backslash c$ ▷ *add c to current group*
    else ▷ *no suitable c and enough group size*
      break ▷ *finalize current group*
  $\mathcal{G} = \mathcal{G} \cup \mathbf{g}$ ▷ *add finalized group*
return $\mathcal{G}$

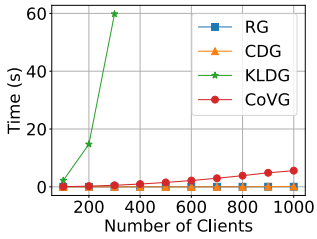


Fig. 5. Running time of different grouping methods.

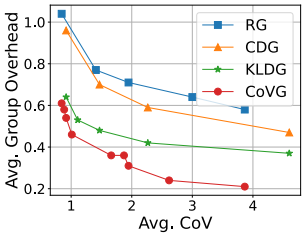


Fig. 6. Average CoV v.s. average group overhead.

Given $|\mathcal{Y}|$ is usually a fixed small number for a given task (e.g., 10 for CIFAR-10 [24], 35 for SpeechCommand (SC) [25]), the time complexity becomes $O(|\mathcal{K}|^3)$. This is cubic to the client number associated with the edge server but irrelevant to the data amount owned by clients.

4.4 Compared with Other Grouping Algorithms

We now compare our algorithm with two existing solutions: clustering then distribution grouping (CDG, used by OUEA [15]) and KLD grouping (KLDG, used by SHARE [16]), as well as random grouping (RG). Fig. 5 shows the time consumed by each algorithm to group different numbers of clients. We can see that RG can group $1,000$ clients at almost no cost (less than 0.3 seconds). CDG has similar efficiency to RG (around 1 second). KLDG is inefficient because i) its time complexity is $O(|\mathcal{K}|^4|\mathcal{Y}|)$; ii) it frequently calculates the KLD, which needs the expensive operation floating-point $log()$. On the contrary, calculating CoV only involves addition and multiplication, which are much cheaper than $log()$. Thus CoVG can group $1,000$ clients in 6 seconds.

We then compare the quality of the grouping results of these four algorithms. Fig. 6 shows a result more directly related to our concerned question: how do different grouping algorithms affect the learning cost and accuracy? With the same cost (i.e., group overhead), CoVG always gives us the best groups with lower CoV (i.e., higher IID degree), which implies better training accuracy. Similarly, to achieve the same level of CoV (i.e., accuracy), CoVG saves us costs. We will present more details about this implication by experiments in Section 6.

Besides, to visualize the differences, as shown in Fig. 7, we use two sets of grouping results of our proposed CoV-Grouping (CoVG) method, compared with RG, CDG, and KLDG. From the first row of Fig. 7 (over CIFAR-10 dataset), Note that most groups generated by RG miss some of the labels; CDG does not significantly improve the grouping as the clustering might be meaningless when data distributions are fundamentally different; KLDG is better than RG and CDG, but CoVG generates the most balanced data within groups ($avgCoV$ is minimized) while keeping group sizes the same. This ensures that when the selected group conducts group training and aggregation, it has the right direction to update its group model. On the contrary, randomly generated groups or groups from unbalanced grouping have more inconsistent objectives with the global model. The second row of Fig. 7 shows the results on the SC dataset, which has more types of data. The observations are similar to those on CIFAR-10. Overall, our CoV-Grouping can lead to better quality (balanced label distribution) of groups compared with other grouping methods.

5 CoV-based Group Sampling

We now discuss our group sampling method, deployed in the cloud as in Fig. 3, which is a probability-based sampling. Each group $\mathbf{g}$ is sampled based on a probability $\mathbf{p}_g$. The key problem is the sampling criteria, i.e., how to compute the sampling probability, which groups should be sampled more frequently, and how frequent it should be. Existing works in FL already considered many sampling criteria to improve the system performance in specific aspects. For example, [26] considers both training time and gradient in sampling to speed up the training. Since now we have the group CoV, it is reasonable to design new sampling methods based on CoV to improve the system performance. In this section, we discuss the possible sampling methods and how to better utilize them. As group heterogeneity widely exists in HFL systems, our designs and observations here are also applicable to other HFL systems.

5.1 Sampling Criteria

Obviously, the probability vector $\mathbf{p}$ should satisfy $\sum_g \mathbf{p}_g = 1$. Based on the sampling probability, our system select groups in each global iteration. Let $\mathcal{S}_t$ be the selected group set in the t-th global iteration.

Similar to our group formation method, our grouping sampling method tends to select those groups whose data are combined IID. Recall that $CoV(\mathbf{g})$ is the CoV of the group $\mathbf{g}$, and a larger $CoV(\mathbf{g})$ means a more biased data distribution in this group. Then, we can compute $\mathbf{p}$ in the following way:

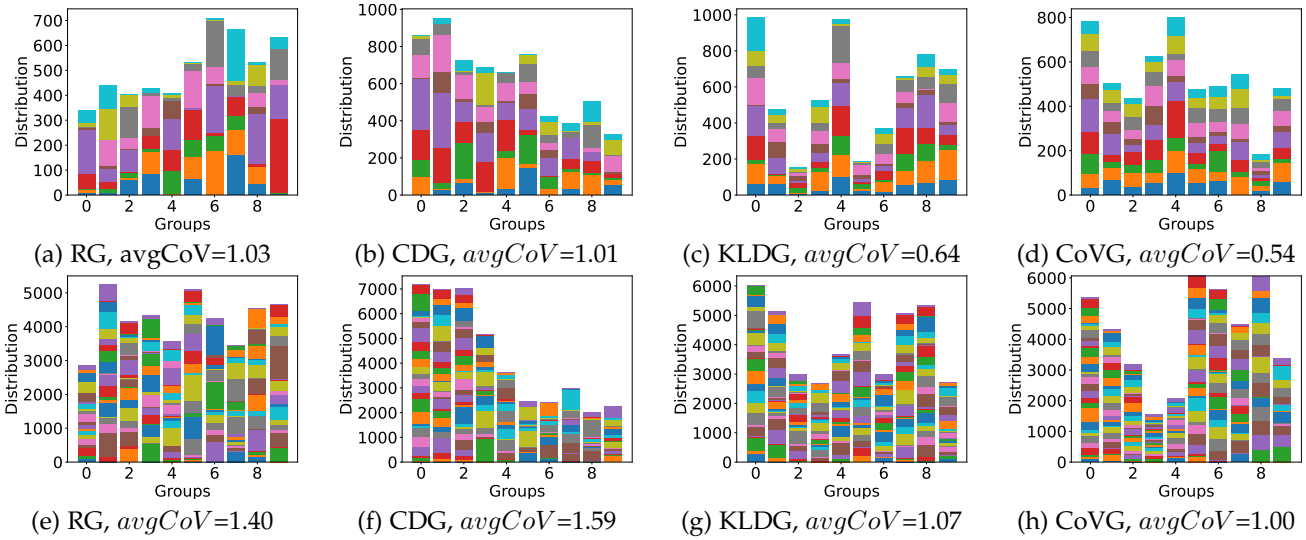
$$p_g = \frac{w(\frac{1}{CoV(\mathbf{g})})}{\sum_{\mathbf{g} \in \mathcal{G}} w(\frac{1}{CoV(\mathbf{g})})}, \tag{38}$$

where $w()$ can be a non-decreasing function. The rationale behind this formula is i) $w()$ reflects the importance of each group but CoV means how bad a group is so we should inverse it in $w()$; p_g is the probability so it is in the format of $w()/\sum w()$ (so all of them sum to 1). We find that the choice of $w()$ also has an impact on the result. We consider three choices[5]: $w(x) = x$, x^2, and e^{x^2}, and use RCoV, SRCoV

5. We choose these three functions, since they amplify the impact of CoV from less to more: the first differs but not much from random selection; the last is close to always selecting the groups with the top CoVs; the middle one is between them. Although one may more elaborately select the function, we simply choose these to show how the learning result varies w.r.t. the sampling function, and our results next confirm that they are at least sufficient for our purpose.

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865



(a) RG, avgCoV=1.03 (b) CDG, $avgCoV$=1.01 (c) KLDG, $avgCoV$=0.64 (d) CoVG, $avgCoV$=0.54

(e) RG, $avgCoV$=1.40 (f) CDG, $avgCoV$=1.59 (g) KLDG, $avgCoV$=1.07 (h) CoVG, $avgCoV$=1.00

Fig. 7. Grouping results and achieved average group CoV ($avgCoV$) of RG, CDG [15], KLDG [16], and CoVG for either CIFAR-10 [(a)-(d)] or Speech Command [(e)-(h)]. Here, each block of color represents one type of data. Blocks represents different types of data even they have the same color. We set $MinGS$=5 for CIFAR-10 and $MinGS$=15 for Speech Command. For CoVG, $MaxCoV$=1.0.

and ESRCoV to denote them. Such methods will be used as SAMPLING-PROB($\mathcal{G}$) in Algorithm 1 to generate $\mathbf{p}$.

Fig. 8 shows the accuracy achieved by these three sampling methods. Overall, the more we emphasize CoV in sampling, the smoother and faster the convergence is. In this set of experiments, we select 12 groups among 60 client groups based on their CoV values in each round. In general, the more we emphasize CoV, the less frequently those groups with larger CoV are sampled, hence less diverse the sampled groups are during the whole training process.

For example, an interesting detail is that in ESRCoV, the top 3 groups occupy more than 92% (top 5 have more than 99%) sampling probability, while in ERCoV the top 6 groups have a probability around 10% each. Thus, ESRCoV only uses 25% client groups (data) to achieve better accuracy than the other methods which uses more client groups.

This seems contradict better performance with more data. But the key reason is that those groups with larger CoV values can have a smaller or even negative impact on convergence as shown in our theoretical analysis. Such results show a similar implication as the second key observation, i.e., more frequently sampled groups tend to dominate the characters of the whole HFL system. Though we yet can not, and it is hard to rigorously confirm this conjecture. This also confirms that the CoV is capable of properly capturing the distribution skew. In our experiments in Section 6, we use ESRCoV sampling as our default CoV sampling method for our methods, since it has the best performance.

5.2 Regrouping Strategy

If we would like to utilize the remaining data in those client groups with larger CoV values, one possible solution is regrouping clients (rerunning the group formation algorithm) after a certain number of global iterations (denoted as regroup interval). In that case, our design of randomly selecting the first client for each group in COV-GROUPING becomes critical and useful. Each time of regrouping can generate different client groups, therefore, giving chances for different clients to be selected during the group sampling. It is obvious that there is trade-off among regroup interval, grouping overhead, and performance gain. We will later show that a smaller regroup interval indeed speeds up convergence by experiments in Section 6.5.

5.3 Handling the Unbiased Factor

As aforementioned in Section 3, Γ_p or $(\frac{1}{p_g})$ can be infinitely large especially when $w()$ amplifies the impact of CoV on sampling and the unbiasedness factor is introduced. Meanwhile, to prioritize the good groups, we hope to assign them a much higher probability (as shown in the comparison of different $w()$). Therefore, when we need to adopt the two mechanisms at the same time, the model is likely to diverge. To avoid catastrophic numerical instability, we will use stabilized aggregation, by normalizing the weights in the following way

$$weight(\mathbf{g}) = \frac{\frac{1}{p_g|\mathcal{S}_t|}\frac{n_g}{n}}{\sum_{g\in\mathcal{S}_t}\frac{1}{p_g|\mathcal{S}_t|}\frac{n_g}{n}}. \tag{39}$$

Such weights will replace $\frac{n_g}{n_t}$ at Line 15 of Algorithm 1. Note that we cannot guarantee that the aggregation is still unbiased after using this normalization. Thus, there is always a trade-off. In addition, when the number of selected groups $|\mathcal{S}_t|$ is close to or even larger than the number of good client groups we have, we will have to select some groups with small p_g, then they will dominate the aggregation because they have large $\frac{1}{p_g}$. Therefore, the selection of $|\mathcal{S}_t|$ needs to be carefully set in practice. This can be done before starting the training, by peeking at the grouping result (the sampling probability), as we always have it prior to training.

6 PERFORMANCE EVALUATIONS

In this section, we report detailed performance evaluations of our proposed GROUP-FEL method via experiments.

6.1 Experiments Setup

Baselines: Upon the selection of baselines, we consider the following three types of related methods: training-based methods (FedProx [7] and SCAFFOLD [8]), grouping methods (CDG from OUEA [15], KLDG from SHARE [16] and QCID from Fed-CBS [21]), and a clustering method for personalized FL (FedCLAR [13]). Classical FedAvg [4] is also included for reference. Note that the reason why we include FedCLAR is to show that personalized FL is not suitable for training a good global model. CDG and KLDG are originally

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

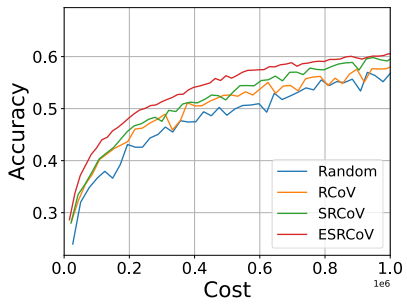


Fig. 8. Different sampling methods: RCoV, SRCoV, ESRCoV & Random.

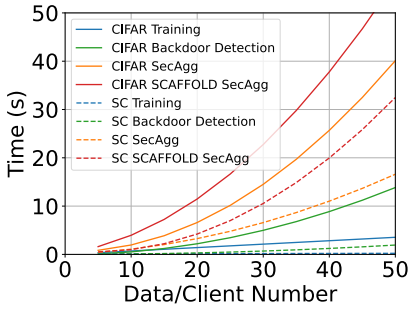


Fig. 9. Overhead measurement over Raspberry PI.

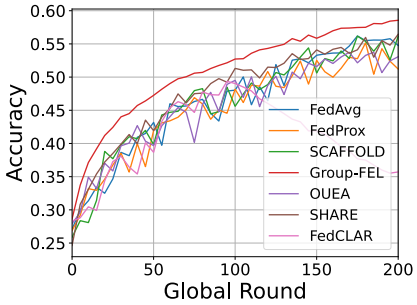


Fig. 10. Accuracy vs iteration - all methods over CIFAR-10.

designed for edge association in HFL, we adopt their basic ideas and port them to group formation algorithms. QCID is based on centralized FL so it originally forms one single subset of all clients. We adopt it into our HFL scenario by letting edge servers sample desired number of groups from their associated clients, separately. Each group is generated based on the QCID sampling method. For a fair comparison, we tune all grouping algorithms so that they tend to generate similar group sizes.

Datasets and ML Models/Tasks: We use CIFAR-10 [24], Fashion-MNIST [27], and Speech Commands (SC) [25] as our training datasets. Fashion-MNIST and CIFAR-10 are popular image classification datasets containing 10 types of pictures. For those dataset, a 1-block and a 3-block ResNet are used to represent light and heavy load tasks vision tasks. The Speech Commands dataset contains 35 types of audio commands and is used for command recognition. For this task, we adopt a 5-layer convolutional neural network (CNN) that is easy to train on RPi to represent lightweight audio tasks. (as shown in Fig. 9). Both image classification and audio recognition are typical edge AI applications.

Total Cost Emulation: As mentioned in Section 2.3, we evaluate the total learning costs based on Equ. (5). To describe the cost of group operations more accurately, we conduct group-based FEL experiments on RPi 4 devices with both CIFAR10 and SC to extract $\mathcal{O}_g()$ and $\mathcal{H}_i()$ according to the collected measurements. Here, all costs are measured by time. As shown in Fig. 9, the original version of secure aggregation (SecAgg) and SCAFFOLD (+SecAgg) [8] are the most costly operation. We use them to estimate different quadratic cost functions for each method and then emulate the costs of group operations in experiments. Other types of group operations can be integrated in the future.

HFL Environment: All training and testing of FL models are performed in a virtual environment developed by our group on a server with 40 cores, 512 GB RAM, 8 $\times$ NVIDIA Tesla V100. The total costs are computed using $\mathcal{O}_g()$ and $\mathcal{H}_i()$ according to the grouping/sampling methods and generated results.

6.2 Performance of Group-FEL

We first test the performance of our design (Group-FEL) with different values of the key hyperparameter $MaxCoV$ and different data heterogeneity (i.e., α). We split CIFAR-10 data to 300 clients with 20 to 200 (normal distribution, restricted by the available data of CIFAR-10) data entries each. On each client, the labels follow the Dirichlet distribution with parameter α[6]. We use three edge servers and each of them has 100 clients.

TABLE 1
Performance of Group-FEL: Group Size (GS), Group CoV, and Accuracy for different α and $MaxCoV$.

α	MaxCoV	GS [min,max](avg)	Avg. CoV	Accu
0.1	0.1	[6, 19](10.96)	0.28	56.68%
	0.5	[5, 11](6.13)	0.43	59.80%
	1.0	[5, 6](5.03)	0.54	**60.56%**
0.5	0.1	[5, 11](7.66)	0.19	64.11%
	0.5	[5, 9](5.23)	0.25	63.40%
	1.0	[5,5](5.00)	0.29	**65.02%**
1.0	0.1	[5, 19](6.95)	0.15	**65.08%**
	0.5	[5, 6](5.02)	0.20	64.85%
	1.0	[5, 5](5.00)	0.20	64.45%

The budget is set as 10^6 unit, which is sufficient for the model to converge. Table 1 shows the detailed performances, including the range and average of group sizes generated, the average group CoV, and the achieved accuracy, when $K = 5$, $E = 2$, and $MinGS = 5$. Clearly, with larger $MaxCoV$ (that allows more skewed distributions), our method generates smaller groups with larger group CoV. Note that smaller group CoV (more balanced data) does not necessarily lead to higher accuracy as it may require larger group sizes and hence higher overhead. When the data is more IID (larger α), smaller $MaxCoV$ leads to better accuracy because we can now have more IID groups with small sizes. However, when the data is skewed, larger $MaxCoV$ may be better. Overall, with less skewed data (larger α), our method can achieve better accuracy.

We then test our method for different group round K and local round E with fixed $\alpha = 0.1$ (so each client mainly has around 1-3 types of data) and $MaxCoV = 0.5$. From Table 2, we can see that although higher K and E generally leads to lower accuracy, which is a nature of FL as suggested by SHARE [16], our method still outperforms hierarchical FedAvg in all cases. Comparison with more baselines are presented in the next subsection.

6.3 Comparison with Existing Methods

Next, we compare our method with the selected baselines, classical FedAvg [4], FedProx [7], SCAFFOLD [8], OUEA [15], SHARE [16], and FedCLAR [13]. For fairness, they are all modified to a hierarchical version (if not originally) with uniform group sampling. FedAvg, FedProx, and SCAFFOLD use random grouping, while FedCLAR uses random grouping at the beginning and then performs its clustering

6. This is adopted by many previous works on non-IID FL, such as [28]. In general, smaller α means more skewed data.

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

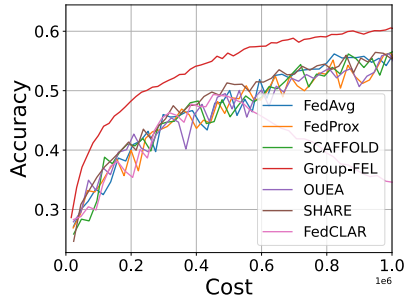


Fig. 11. Accuracy vs cost - CIFAR10.

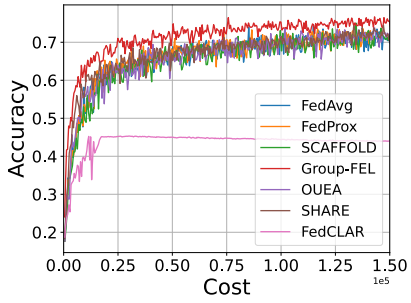


Fig. 12. Accuracy vs cost - Fashion-MNIST.

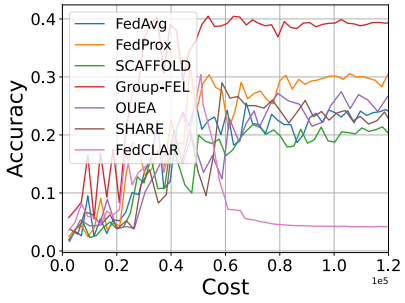


Fig. 13. Accuracy vs cost - SC.

TABLE 2
Accuracy of Group-FEL with different K and E.

K	E	Group-FEL	FedAvg
5	2	59.0%	55.3%
	5	59.1%	54.2%
	10	56.7%	54.5%
10	2	59.3%	55.3%
	5	57.8%	54.7%
	10	56.5%	52.7%
20	2	58.8%	53.5%
	5	55.6%	54.0%
	10	51.5%	50.1%

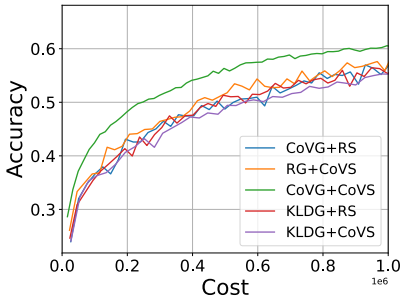


Fig. 14. Different grouping and sampling methods.

method at a specific round. OUEA uses its CDG algorithm (Algorithm 1 in [15]) and SHARE uses its KLD-based grouping. QCID is not included here since it inherently creates and samples different groups in each round. Later, it will be compared with our re-grouping method in Section 6.5.

Fig. 10 shows the results of accuracy over global iterations on CIFAR-10. We can see that our method outperforms all baselines while the baselines do not differ much from each other. Note that the accuracy of FedCLAR drops after clustering since it is designed for personalized FL and is not suitable for training the global model. Fig. 11 shows the same results over the corresponding training cost. Clearly, our method advances even more in this measurement. With the same training cost, our method can achieve significantly higher accuracy. The reason is that FedProx and SCAFFOLD demand more computation (both) and communication (SCAFFOLD) in each round; OUEA and SHARE, even though we tune their group size, still generate some costly groups as they do not control the group size. Compared with Fig 10, Fig. 11 can illustrate the critical advance of our proposed method over existing FL solutions more clearly.

Fig. 12 shows the results from similar experiments over the Fashion-MNIST. All the baselines behave similarly on Fashion-MNIST as they do on CIFAR-10. Two slight difference are i) Fashion-MNIST is a simpler task so the difference between our method and others are not that large now; ii) the accuracy of FedCLAR drops much slower after clustering. We also conduct similar experiments over the Speech Command (SC) dataset, in which there are 35 types of commands. We set $\alpha = 0.01$, which means the data on each client is extremely skewed: the data on each client are mainly dominated by less than 5 types of data. We set $MinGS = 15$ for all and no $MaxCoV$ constraint. Fig. 13 shows the results. Clearly, the convergence is unstable due to the serious inconsistency (large ζ). In general, we can observe similar results as those on CIFAR-10.

6.4 Impacts of Group Formation and Group Sampling

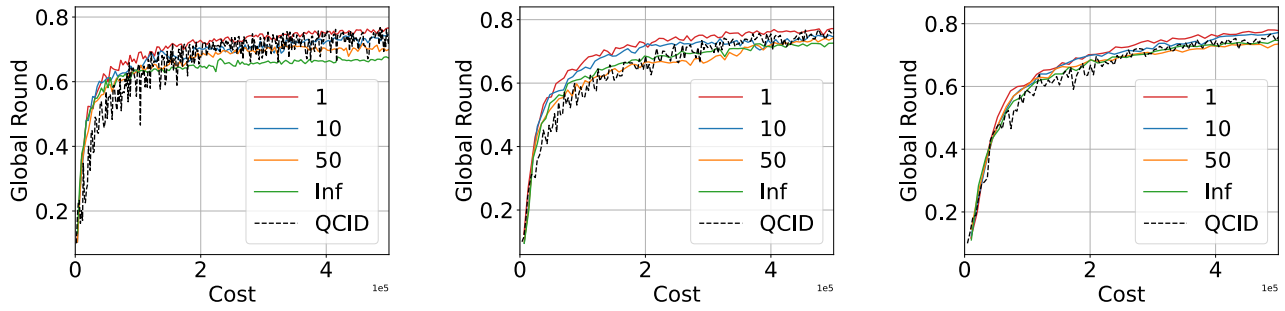
Next, we investigate the impacts of group formation and group sampling in our proposal methods. Fig. 14 shows different combinations of group formation methods (random grouping (RG), KLD-based grouping (KLDG), and our proposed CoV grouping (CoVG)) and group sampling methods (random sampling (RS) and our proposed CoV sampling (CoVS)). CDG is omitted as it does not show a significant difference from RG. The key observation from this result is the advantage of the proposed methods is more clear when both CoVG and CoVS are used together. When only CoVG is adopted and random sampling is used, the good groups are not prioritized so they do not have much impact on the learning result. Compared with the performance of our method (CoVG+CoVS), we can see CoVS indeed adding a significant advantage over CoVG. When CoVS is adopted alone, the quality of prioritized groups is not fundamentally better than others due to poor grouping. We repeat this set of experiments on the SC dataset and observe similar results. Therefore, our recommendation is to use both CoVG and CoVS as we did in our GROUP-FEL.

6.5 Impacts of Re-Grouping

Finally, as discussed in Section 5.2, regrouping helps to speed up convergence, thus we evaluate our regrouping strategy as well. Fig. 15 shows the results of regrouping when there are 100, 200, and 300 clients in the system, respectively. Fig. 15(a) clearly shows that a smaller re-group interval leads to faster convergence. However, smaller re-group interval also add more grouping overhead. Fig. 15(b) and Fig. 15(c) show similar results, but in comparison, we can see that when there are more clients (hence more groups) available, the benefit of re-grouping decreases. This is because when there are fewer data available, the diversity is critical for the model performance. On contrary, when the number of clients is large, the data is already diverse enough even without re-grouping and re-selection. In addition, we

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865



(a) With 100 clients (b) With 200 clients (c) With 300 clients

Fig. 15. Impacts of re-grouping (vs QCID) with different re-group intervals (curves with labels of 1, 10, 50 and Inf (i.e., without re-grouping)) and different number of clients (i.e., (a) 100, (b) 200, (c) 300).

also plot the grouping method based on QCID sampling in Fig. 15. Compared to our method (especially interval=1), QCID is not as stable and generally performs slightly worse.

7 Related Works

7.1 Non-IID in Federated Learning

Recently, the Non-IID problem in FL has been well-studied in prior research such as [6]–[11], [13], [14], [29]. These works can be categorized into two types based on how they address statistical heterogeneity and their corresponding objectives. The first type of work, such as [6]–[8], aims to reduce the negative impact of objective inconsistency caused by skewed data and obtain a global model that performs well in the overall task (where there is no data distribution skew). The other type of work recognizes that heterogeneity reflects the different natures and demands of different clients, thus they focus on training multiple personalized models designed for different users, instead of training a good global model. This kind of method is also called *personalized FL*. In general, if the goal is to extend the available data to generate a general model, the first approach may be used. If the goal is to improve the immediate user experience, the personalized approach may be more suitable.

To train a global model that works well for the overall task, we need to strive to remedy the negative impact of inconsistent local objectives. The pioneer work mainly uses training-based methods, such as augmenting the data, modifying the loss function, and changing the descent direction. Zhao *et al.* [6] utilize a globally shared dataset to bootstrap and reduce the label distribution skew among clients. FedProx [7] limits the divergence of local training from the last global model to mitigate inaccurate updates. SCAFFOLD [8] records the direction of local and global gradient to re-direct updates to an estimated correct direction. IGFL [30] leverages both individual and group updates to mimic the distributions and digests them in both client and server optimization, thereby improving the ability to deal with heterogeneity. CCVR [12] calibrates the classifier layer using sampled virtual representations, while DFL [31] tries to separate global and local-only features to perform alternative local-global optimization for both generalized adaption and personalized performance. Fed-CBS [21] also proposes a heterogeneity-aware client sampling mechanism to reduce class-imbalance of the grouped dataset from the intentionally selected clients aiming to facilitate convergence. For personalized FL, as suggested by [32], there are many methods, including but not limited to transfer learning [13], meta-learning [33], knowledge distillation [34], and clustering [13].

This paper focuses on the first type of approach to train a global model and our proposed method outperforms existing methods in the HFL environment. We choose the two most popular methods FedProx and SCAFFOLD as part of the baselines in experiments. Although personalized FL methods do not align well with our target scenario, we also include FedCLAR [13] in experiments to confirm that this type of solution is not suitable in our considered scenario.

7.2 Hierarchical Federated Learning (HFL)

HFL has been studied recently due to advances in the scalability and privacy protection of FL. A cloud-edge-client HFL framework has been studied in [2] and it gives the convergence analysis of HFL. Wang *et al.* [35] also considered cluster structure formation in HFL where edge servers are grouped in different clusters for model aggregation. While both [2] and [35] provide the HFL convergence analysis, their analysis is different from ours, since they do not consider (1) group sampling (i.e., the sampling fraction for each group) and (2) group characters (such as the variance of data number among clients and groups).

To handle the non-IID problem in HFL, OUEA [15] and SHARE [16] consider the client assignment problem: they try to assign clients to edge servers such that the clients associated with each edge server have a balanced data distribution when virtually combined. OUEA [15] first clusters similar clients together and then distributes them to different groups so that the data distribution inside each group tends to be IID. OUEA also offers a convergence analysis, but similar to [2], they do not consider group sampling and some group-specific characters. Furthermore, OUEA requires the loss functions to be convex (which is not true for neural networks) while our analysis does not. SHARE [16] also considers data distribution among edge aggregators and optimizes communication cost during the client-edge assignment. It uses Kullback–Leibler divergence (KLD) to measure the IID degree among edge aggregators. Both OUEA and SHARE consider each edge server as one single aggregator (i.e., aggregating one client group) and do not control the group size. In the experiments, we port their assignment policy to the group formation and re-implement their algorithms for comparison with our grouping method. FedGroup [36] also considers a group-based FL where clients are grouped based on their local model similarities.

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

It constructs a data-driven distance measure to learn the proximities between clients' local optimizations and then determine the grouping and the goals of each group.

All of the works above consider the hierarchical structure and/or the communication cost, but none of them takes into account the *overhead incurred by group operations* (especially those for security and privacy protection operations). As a result, they do not limit the number of clients associated with an edge aggregator (within a group), which may not be cost-efficient, as shown in Fig. 2. Additionally, some efforts focused on privacy protection and participant selection in HFL. Wainakh *et al.* [37] pointed out the gain of HFL on privacy enhancement. Yang *et al.* [38] proposed a compression mechanism and gradient clipping method in an HFL architecture to reduce communication overhead and protect privacy. Wei *et al.* [39] studied the participant selection problem of a multi-model HFL. Our work instead focuses on group formation and sampling in a group-based HFL to reduce the total learning cost (including overhead caused by group operations).

7.3 Performance Measurement

Most existing works measure convergence by iteration, which is not effective in many cases. For example, some algorithms [7], [8] require more computation and/or communication in each round to achieve faster convergence regarding global iterations but may be slower when measured by wall clock time and/or resource cost. Therefore, more realistic measurements for FL systems have been investigated to satisfy different application requirements. Luo *et al*. [26] seek to reduce the training wall clock time. They propose a new convergence upper bound for arbitrary client selection probabilities and generate a non-convex training time minimization problem. Their approach significantly reduces the convergence time to achieve the same target loss compared to several baselines regarding the wall-clock time. Yang *et al*. [40] study the energy consumption optimization problem for battery-sensitive devices and the proposed method can save up to 59.5% energy. Some works [41], [42] notice that communication traffic may also be a potential bottleneck for cross-device FL systems, and hence seek to reduce the bandwidth requirement by gradient/model compression. They compare the convergence rates by loss over total network traffic. There are also works [43]–[47] focusing on minimizing both training and communication costs for training multiple FL models in a shared edge cloud. In this paper, we consider all costs incurred by training and group operations and unify the latter into one quadratic function. We have evaluated the performance of FL systems in terms of accuracy over this generally defined cost.

8 Conclusion

In this paper, we first tackle the challenge of group formation in group-based HFL, which is critical due to the widely adopted group operations (for privacy and security) and yet remains unresolved. Particularly, we demonstrate through both theoretical analysis and empirical results that the group size and group data distribution are key factors for the group formation in group-based HFL and have a significant impact on its convergence and total cost. To address this, we design a greedy grouping algorithm based on group CoV to reduce group size while maintaining the relatively IID group data. In addition, group sampling methods for CoV-aware groups are also proposed and analyzed. Through extensive experiments, we show that current popular FL algorithms do not perform well in the context of HFL, and our methods outperform them across various scenarios. This work contributes valuable insights and practical solutions to the often-overlooked challenges associated with group-based hierarchical federated learning at edge systems. We believe that our solution can support more intelligent applications through edge computing and mobile AI.

Looking ahead, our future work aims to enhance the proposed group formation and sampling strategies. This involves exploring ways (1) to incorporate the parameter γ, related to the CoV of data sample amounts among clients, (2) to ensure fairness among clients and data representation within the framework of group-based HFL. This continued exploration is anticipated to further refine the efficacy and fairness of our proposed methods, offering advancements in the field of hierarchical federated learning.

References

[1] J. Liu, X. Wei, X. Liu, H. Gao, and Y. Wang, "Group-based hierarchical federated learning: Convergence, group formation, and sampling," in *Proc. of ICPP*, 2023.

[2] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proc. of IEEE ICC*, 2020.

[3] K. Bonawitz, *et al.*, "Towards federated learning at scale: System design," *Proc. of MLSys*, 2019.

[4] B. McMahan, *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *Proc. of AISTATS*, 2017.

[5] K. Bonawitz, *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. of ACM CCS*, 2017.

[6] Y. Zhao, M. Li, *et al.*, "Federated learning with non-IID data," *arXiv preprint arXiv:1806.00582*, 2018.

[7] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proc. of MLSys*, 2020.

[8] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *Proc. of ICML*, 2020.

[9] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," *arXiv arXiv:2002.04758*, 2020.

[10] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," *arXiv preprint arXiv:1802.07876*, 2018.

[11] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," *arXiv preprint arXiv:1906.06629*, 2019.

[12] M. Luo, *et al.*, "No fear of heterogeneity: Classifier calibration for federated learning with non-IID data," *Proc. of NeurIPS*, 2021.

[13] R. Presotto, G. Civitarese, and C. Bettini, "Fedclar: Federated clustering for personalized sensor-based human activity recognition," in *Proc. of IEEE PerCom*, 2022.

[14] Y. Li, F. Li, *et al.*, "Power of redundancy: Surplus client scheduling for federated learning against user uncertainties," *IEEE Trans. on Mobile Computing*, vol. 22, no. 9, pp. 5449–5462, 2023.

[15] N. Mhaisen, *et al.*, "Optimal user-edge assignment in hierarchical federated learning based on statistical properties and network topology constraints," *IEEE Trans. on Network Science and Engineering*, vol. 9, no. 1, pp. 55–66, 2021.

[16] Y. Deng, F. Lyu, *et al.*, "SHARE: Shaping data distribution at edge for communication-efficient hierarchical federated learning," in *Proc of IEEE ICDCS*, 2021.

[17] T. D. Nguyen, P. Rieger, *et al.*, "Flame: Taming backdoors in federated learning," *Cryptology ePrint Archive*, 2021.

[18] H. Yang, M. Fang, and J. Liu, "Achieving linear speedup with partial worker participation in non-IID federated learning," *Proc. of ICLR*, 2021.

[19] L. Wang, Y. Guo, T. Lin, and X. Tang, "Client selection in non-convex federated learning: Improved convergence analysis for optimal unbiased sampling strategy," *arXiv arXiv:2205.13925*, 2022.

[20] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. of ICLR*, 2019.

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Cloud Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCC.2024.3482865

[21] J. Zhang, A. Li, *et al.*, "Fed-CBS: A heterogeneity-aware client sampling mechanism for federated learning via class-imbalance reduction," in *Proc. of ICML*, 2023.

[22] M. Mahajan, *et al.*, "The planar k-means problem is NP-hard," *Theoretical Computer Science*, vol. 442, pp. 13–21, 2012.

[23] D. Aloise, *et al.*, "NP-hardness of Euclidean sum-of-squares clustering," *Machine learning*, vol. 75, pp. 245–248, 2009.

[24] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," Technical Report, U. of Toronto, 2009.

[25] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," *ArXiv e-prints*, Apr. 2018.

[26] B. Luo, W. Xiao, S. Wang, J. Huang, and L. Tassiulas, "Tackling system and statistical heterogeneity for federated learning with adaptive client sampling," in *Proc. of IEEE INFOCOM*, 2022.

[27] H. Xiao, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *arXiv arXiv:1708.07747*, 2017.

[28] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.

[29] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *Proc. of NeurIPS*, 2020.

[30] H. Huang, F. Shang, Y. Liu, and H. Liu, "Behavior mimics distribution: Combining individual and group behaviors for federated learning," in *Proc. of IJCAI*, 2021.

[31] Z. Luo, Y. Wang, *et al.*,"Disentangled federated learning for tackling attributes skew via invariant aggregation and diversity transferring," *arXiv preprint arXiv:2206.06818*, 2022.

[32] A. Z. Tan, *et al.*, "Towards personalized federated learning," *IEEE Trans. on Neural Networks and Learning Systems*, 2022.

[33] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," *arXiv preprint arXiv:2002.07948*, 2020.

[34] J. Zhang, S. Guo, X. Ma, H. Wang, W. Xu, and F. Wu, "Parameterized knowledge transfer for personalized federated learning," *Proc. of NeurIPS*, 2021.

[35] Z. Wang, H. Xu, J. Liu, H. Huang, C. Qiao, and Y. Zhao, "Resource-efficient federated learning with hierarchical aggregation in edge computing," in *Proc. of IEEE INFOCOM*, 2021.

[36] M. Duan, D. Liu, *et al.*, "FedGroup: Efficient federated learning via decomposed similarity-based clustering," in *Proc. of IEEE ISPA/BDCloud/SocialCom/SustainCom*, 2021.

[37] A. Wainakh, A. S. Guinea, T. Grube, and M. Mühlhäuser, "Enhancing privacy via hierarchical federated learning," in *Proc.of IEEE EuroS&PW*, 2020, pp. 344–347.

[38] H. Yang, "H-FL: A hierarchical communication-efficient and privacy-protected architecture for federated learning," *arXiv preprint arXiv:2106.00275*, 2021.

[39] X. Wei, J. Liu, X. Shi, and Y. Wang, "Participant selection for hierarchical federated learning in edge clouds," in *Proc. of IEEE NAS*, 2022.

[40] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. on Wireless Communications*, vol. 20, no. 3, pp. 1935–1949, 2020.

[41] J. Hamer, M. Mohri, and A. T. Suresh, "FedBoost: A communication-efficient algorithm for federated learning," in *Proc. of ICML*, 2020.

[42] H. Gao, A. Xu, and H. Huang, "On the convergence of communication-efficient local sgd for federated learning," in *Proc. of AAAI*, vol. 35, no. 9, 2021, pp. 7510–7518.

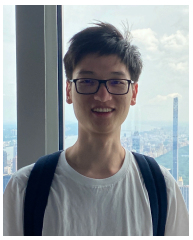
[43] X. Wei, J. Liu, and Y. Wang, "Joint participant selection and learning scheduling for multi-model federated edge learning," in *Proc. of IEEE MASS*, 2022.

[44] X. Wei, J. Liu, and Y. Wang, "Joint participant selection and learning optimization for federated learning of multiple models in edge cloud," *J. of Computer Science and Technology*, vol. 38, no. 4, pp. 754–772, 2023.

[45] X. Wei, L. Fan, *et al.*, "Quantum assisted scheduling algorithm for federated learning in distributed networks," in *Proc. of IEEE ICCCN*, 2023.

[46] X. Wei, L. Fan, *et al.*, "Hybrid quantum-classical benders' decomposition for federated learning scheduling in distributed networks," *IEEE Trans. on Network Science and Engineering*, to appear.

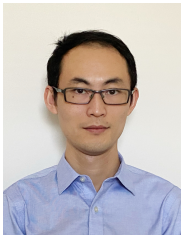
[47] X. Wei, K. Ye, *et al.*, "Joint Participant and Learning Topology Selection for Federated Learning in Edge Clouds," *IEEE Trans. on Parallel and Distributed Systems*, vol. 35, no. 8, pp. 1456-1468, 2024.



Jiyao Liu is a Ph.D. student at the Department of Computer and Information Sciences at Temple University. He received his B.E. degree in Information Security from North China University of Technology in 2020. His research interests include AI, security, and edge computing.

Xuanzhang Liu received the master degree in computer science from University of Delaware in 2020. He is currently pursuing his PhD degree at the Department of Computer and Information Science, Temple University. His main research interests include edge computing, blockchain, and federated learning.

Xinliang Wei (S'21-M'23) is an Assistant Professor at Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences. He holds a Ph.D. in Computer and Information Sciences from Temple University, USA in 2023. He received his M.S. and B.E. degrees both in Software Engineering from SUN Yat-sen University, Guangzhou, China in 2016 and 2014, respectively. His research interests include edge computing, federated learning, reinforcement learning, and Internet of Things. He is a recipient of Outstanding Research Assistant from College of Science and Technology and Scott Hibbs Future of Computing Award from Department of Computer & Information Sciences at Temple University.



Hongchang Gao is an Assistant Professor in the Department of Computer and Information Sciences at Temple University. He holds a Ph.D. in Computer Engineering from University of Pittsburgh, an MEng in Computer Science from Beihang University, and a B.S. in Mathematics and Applied Mathematics from Ocean University of China. His research interests include machine learning, optimization, and data mining. He was selected for AAAI 2023 New Faculty Highlights, and has served as Associate Editor for Journal of Combinatorial Optimization.

Yu Wang (S'02-M'04-SM'10-F'18) is a Professor and Chair of the Department of Computer and Information Sciences at Temple University. He holds a Ph.D. from Illinois Institute of Technology, an MEng and a BEng from Tsinghua University, all in Computer Science. His research interest includes wireless networks, smart sensing, and mobile computing. He has published over 300 papers in peer reviewed journals and conferences. He is a recipient of *Ralph E. Powe Junior Faculty Enhancement Awards* from Oak Ridge Associated Universities (2006), *Outstanding Faculty Research Award* from College of Computing and Informatics at the University of North Carolina at Charlotte (2008), *Fellow of IEEE* (2018), and *ACM Distinguished Member* (2020). He has served as Associate Editor for *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Cloud Computing*, among others.

Authorized licensed use limited to: Temple University. Downloaded on November 17,2024 at 00:32:04 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.