FedALL: Minimizing Variance of Partial Participation in Federated Learning

Zeyuan Zuo, Yi Zeng, Jason Pui Yin Cheung, Baochun Li, *Fellow, IEEE*, and Teng Zhang, *Senior Member, IEEE*

***Abstract*—In federated learning, partial participation, where offline or delayed clients are absent, can significantly degrade model performance. A potential solution is to better utilize updates from these absent clients. Existing methods typically cache the most recent clients' updates and reuse them in subsequent training rounds, but they often lack a clear theoretical justification for why they should use these updates and how to optimally incorporate them in model training. In this study, we show that leveraging the most recent updates can reduce variance in model aggregation, providing insights into the optimal strategy for using these updates. In particular, we prove that selecting the appropriate decay coefficient for cached updates minimizes variance in aggregation. Building on this insight, we propose FedALL, a novel federated learning algorithm designed to minimize aggregation variance. Extensive experiments across different datasets, client participation ratios, and levels of data heterogeneity indicate that FedALL consistently outperforms FedAvg by an average increase of 2.51% in accuracy and an average decrease of 0.74% in standard deviation. Compared to the previous state-of-the-art method, FedALL demonstrates a 1.85% improvement in accuracy and a 0.6% improvement in standard deviation.**

***Index Terms*—Federated learning, optimization, partial participation, variance reduction.**

I. Introduction

FEDERATED learning is a distributed learning paradigm where multiple devices collaboratively train a shared model without exchanging raw data [1]. In practice, full participation, where all devices engage in every communication round, is rarely feasible due to constraints such as limited network bandwidth, device battery life, and computational resources. Instead, partial participation, where only a subset of devices contribute to the aggregation in each round, is more common in real-world scenarios [2], [3], [4], [5].

To formalize these concepts, we use t to denote the index of a communication round, and t_e to represent the deadline set by the server for round t. In each round, the server distributes the global model to clients, which then perform local training and return their updates.

Received 12 January 2026; revised 22 March 2026 and 11 May 2026; accepted 27 May 2026. Date of publication 1 June 2026; date of current version 10 August 2026. This work was supported in part by the National Natural Science Foundation Young Scientists Fund under Grant 82303957 and in part by the Health and Medical Research Fund (HMRF) under Grant 19200911. *(Corresponding author: Teng Zhang.)*

Zeyuan Zuo, Yi Zeng, Jason Pui Yin Cheung, and Teng Zhang are with the Department of Orthopaedics and Traumatology, The University of Hong Kong, Hong Kong, SAR, China (e-mail: zeyuanz@connect.hku.hk; yizeng@connect.hku.hk; cheungjp@hku.hk; tgzhang@hku.hk).

Baochun Li is with the School of Biomedical Engineering, The University of Hong Kong, Hong Kong, SAR, China (e-mail: bli@ece.toronto.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/JIOT.2026.3698499, provided by the authors.

Digital Object Identifier 10.1109/JIOT.2026.3698499

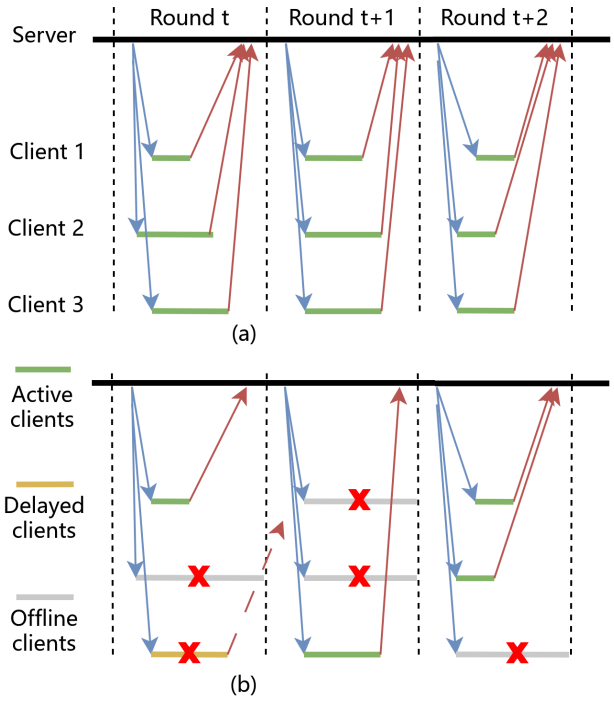

Fig. 1. Illustration of (a) full participation and (b) partial participation.

A client i is considered active in round t if it has the capability to receive the global model and send the model update to the server before the deadline t_e. On the contrary, a delayed client fails to complete training and return its update by t_e. In a communication round, each client is either online or offline, and an online client is further classified as active or delayed. The server selects a subset of clients from a pool consisting of all online clients. Full participation is defined as the scenario where all clients are active in every round. Partial participation occurs when only a subset of clients is active in each round, due to factors such as clients being offline or delayed. The difference between full participation and partial participation is illustrated in Fig. 1.

Partial participation introduces significant challenges, such as slower convergence, increased training instability, and reduced model performance [1], [6], [7], [8]. These issues arise because active clients may only possess low-quality or highly biased data, leading to biased model aggregation. The bias then propagates to subsequent rounds, as the aggregated model serves as the starting point for the next round of training. Consequently, the accumulation of bias results in suboptimal model performance.

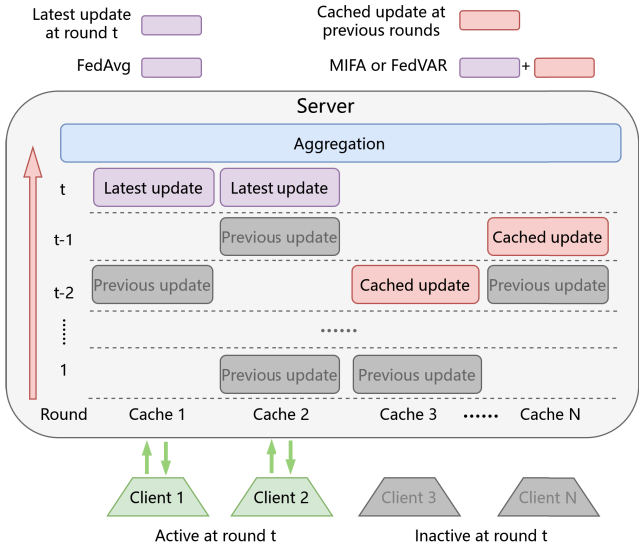

Fig. 2. Architecture of federated learning methods that only use the latest updates of active clients (FedAvg) and methods using clients' most recent updates (e.g., FedVARP and MIFA).

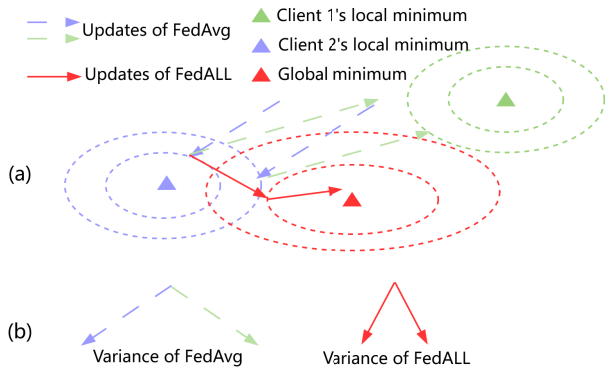

Fig. 3. (a) Example of federated learning where two clients participate alternately. FedALL can better approach the global minimum than FedAvg. (b) Variance of each update of FedAvg and FedALL during training. FedALL has a smaller variance.

There are two primary solutions to these challenges. The first solution involves client selection algorithms, which aim to select a subset of clients from the active pool for training [9], [10], [11], [12]. However, these methods may not be directly applicable to partial participation scenarios, as the active pool could predominantly consist of clients with low-quality or highly biased data, thereby limiting the effectiveness of such selection strategies [13].

The second solution focuses on incorporating contributions from inactive clients into the model aggregation process. This is typically achieved by caching updates from inactive clients and reusing them in future rounds. These cached updates may contain valuable information, which can help mitigate bias during aggregation [6], [7], [14], [15]. Notably, most of these approaches adopt a common strategy: using clients' most recent updates as surrogate updates for inactive clients [6], [7], [14], [15]. This framework is illustrated in Fig. 2. In each training round, active clients send their updates to the server. The server then caches these updates and uses them for model aggregation in future rounds.

The motivation for using clients' most recent updates is straightforward. When a client is unavailable, its update is absent from the aggregation process in the current round, leading to biased model aggregation. By caching updates, the server can reuse this information to mitigate bias and improve the aggregation process. On the flip side, since cached updates are generated in previous rounds and have already been incorporated into prior aggregations, they may become outdated in the current round. This staleness can potentially result in ineffective or even detrimental model aggregation. Moreover, cached updates require substantial storage on the server, which poses a practical challenge, especially when the number of clients is large.

Although the intuition is clear, existing work has not provided a theoretical perspective or analysis to quantitatively explain *the potential benefits of using cached updates and the optimal way to use them*. In this article, we address these gaps by proposing a theoretical framework that analyzes the bias and variance in the model aggregation process under partial participation. We demonstrate that leveraging clients' most recent updates can reduce variance in the aggregation process, which is the key benefit of this approach. In particular, we prove that using these updates transforms the variance of the aggregated model across clients into variance across rounds for each client, thereby highlighting the potential for variance reduction in the aggregation process.

Building on this insight, we propose a new algorithm, FedALL, which fully exploits the benefits of variance reduction. The core idea is to assign a decay coefficient to cached updates, regulating their contributions to the aggregation process. The decay coefficient is adaptive, with its value dynamically adjusted based on the training process. We theoretically prove that, by choosing the appropriate decay coefficient, the variance of the aggregated model can be minimized. To improve efficiency in storage-constrained scenarios, we design a storage-efficient module for FedALL. It reduces the storage complexity from $\mathcal{O}(N)$ to $\mathcal{O}(rM)$, where N and M are the total number of clients and active clients. r is a hyperparameter typically much smaller than N/M.

We illustrate the benefit of variance reduction through Fig. 3. In this example, two clients participate alternately, and FedAvg exhibits a zigzag pattern between the local minima of the two clients. On the contrary, FedALL demonstrates more effective convergence to the global minimum.

We conducted extensive experiments to evaluate FedALL and other state-of-the-art federated learning algorithms. The results indicate that FedALL consistently outperforms existing methods across various datasets, client participation ratios, and levels of data heterogeneity. In particular, FedALL with the second approach for estimating the decay coefficient exhibits greater robustness and higher accuracy.

Our contributions are summarized as follows. *First*, we propose FedALL, a novel algorithm that effectively reduces variance in model aggregation. *Second*, we propose a theoretical framework to analyze the bias and variance of model aggregation under partial participation to explain the benefits of FedALL. *Third*, we demonstrate the effectiveness of FedALL through extensive experiments across different settings. The results show that FedALL improves accuracy by an average of 2.51% and reduces standard deviation by an average of 0.74% when compared to FedAvg across 30 different

tasks. Moreover, FedALL achieves a 1.85% improvement in accuracy and a 0.6% improvement in standard deviation when compared to the previous state-of-the-art method, FedVARP.

II. Related Work

A. Possible Solutions to Partial Participation

Federated learning, first introduced by [1], highlighted several challenges including partial participation [4], [5], [16]. As previously mentioned, client selection strategies can mitigate this issue by optimizing the utilization of active clients based on device availability and data heterogeneity.

For instance, FedCS employs a greedy algorithm to select as many clients as possible within a specified deadline [10]. POWER-OF-CHOICE prioritizes clients with higher local training loss and demonstrates that biased selection can accelerate convergence [9]. FOLB selects clients based on the correlation between their local updates and the global update, using this as an estimation of their contributions [11].

However, client selection algorithms primarily focus on utilizing active clients, which can lead to suboptimal performance when the pool of active clients is limited or when these clients fail to represent the entire client population. Furthermore, identifying the optimal number of clients to select poses a challenge, particularly in scenarios characterized by high levels of data heterogeneity. In such cases, achieving an effective balance between efficiency and convergence rate becomes critically important [13].

Another strategy focuses on optimizing the utilization of inactive clients. In these approaches, updates from all clients are cached and reused when clients are inactive in future rounds. MIFA introduces this perspective by averaging updates from active clients and cached updates from inactive clients, weighted by the number of samples per client [6]. However, MIFA assumes full participation in the first communication round, which is often impractical in real-world scenarios. To address this limitation, FedVARP uses cached updates from clients to compute two vectors, which are then used for model aggregation [7]. This approach eliminates the client participation error term in the convergence analysis.

A significant drawback of these methods is their substantial storage requirements on the server, scaling at $\mathcal{O}(N)$. Here N is the total number of clients. On the contrary, FedAU presents a lightweight framework that does not cache updates but instead adjusts coefficients for each active client based on their participation histories [17]. This approach reduces the storage complexity to $\mathcal{O}(M)$, where M is the number of active clients.

B. Variance Reduction

Variance reduction is a widely recognized technique in stochastic optimization that enhances convergence rates and optimization stability [18], [19], [20]. This approach has been successfully integrated into federated learning to achieve similar objectives [7], [21], [22]. SCAFFOLD was the first to employ variance reduction to mitigate the variance caused by data heterogeneity [21]. It maintains a global control variate on the server to adjust biased local gradients on the client side. This global control variate represents the average of all uploaded local control variates, which capture client drifts. Despite sharing the same model aggregation step as FedAvg, SCAFFOLD introduces an additional control variate aggregation step. It requires clients to upload and download both models and control variates, doubling the communication overhead. To address this limitation, FedPVR improves upon SCAFFOLD by applying variance reduction to only a subset of model layers, thereby reducing the communication overhead [22].

Inspired by SAGA, FedVARP employs variance reduction to tackle the issue of partial participation [7], [18]. It adapts the SAGA algorithm for use in the model aggregation step, with theoretical analysis demonstrating that FedVARP eliminates the partial participation error term. While the analysis confirms convergence, it provides limited insight into performance improvements. Furthermore, the convergence time complexity of FedVARP remains $\mathcal{O}(1/\sqrt{T})$, identical to that of FedAvg, despite eliminating the partial participation error term. The convergence analysis is also subject to diverse assumptions. Typically, convergence time complexity with respect to the number of training rounds is used to evaluate efficiency. When methods share the same complexity, but under different assumptions, determining which method performs better becomes challenging.

On the other hand, variance can help escape local minima and improve the generalization of the model. While our work focuses on minimizing the variance in model aggregation, it does not completely eliminate it because variance arises from both the model aggregation and the local training process. By retaining variance from local training and reducing that from model aggregation, we maintain the benefits of variance, such as escaping local minima while enhancing the stability of model performance under varying device availability patterns.

III. Problem Setup and Proposed Algorithm

In this section, we first mathematically define the optimization objective of the partial participation problem in terms of the bias and variance during the aggregation step. Next, we present our proposed algorithm, FedALL, which is designed to minimize the variance introduced by partial participation. Finally, we discuss an additional storage-efficient module for FedALL, making it more practical for servers with limited storage capacity.

A. Problem Setup

We begin by defining some notations. Let S denotes the set of all clients with $|S| = N$, and let S_t denote the set of clients participating in the tth round with $|S_t| = M$. The model update of client i at tth round is defined as $\Delta^i_{w,t} = w^i_t - w^i_{t-1}$. We use t'_i to represent the most recent active round of client i before round t.

In the context of partial participation, the aggregation step can be viewed as an *estimator* of the aggregation step under full participation. Taking FedAvg as an example, the aggregation steps under full participation and partial participation can be expressed as (a) and (b), respectively

$$(a)\ \Delta_{w,t} = \frac{1}{N}\sum_{i\in S}\Delta^i_{w,t}, \quad (b)\ \widehat{\Delta}_{w,t} = \frac{1}{M}\sum_{i\in S_t}\Delta^i_{w,t}.$$

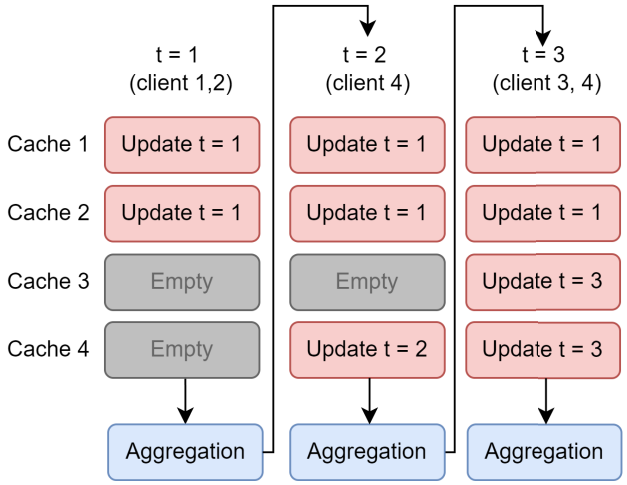

Fig. 4. Overview of the FedALL algorithm.

Here, $\Delta_{w,t}$ represents the desirable aggregation result, which can only be computed when all clients participate. On the contrary, $\widehat{\Delta}_{w,t}$ represents the aggregation result under partial participation, and serves as an estimate of $\Delta_{w,t}$.

We can use bias and variance to evaluate the quality of this estimator. Bias measures the systematic error, while variance captures the variability across different participation patterns. They are defined as follows:

$$\text{Bias} = \mathbb{E}_{S_t}\left[\widehat{\Delta}_{w,t}\right] - \Delta_{w,t} \tag{1}$$

$$\text{Variance} = \mathbb{E}_{S_t}\left[\left\|\widehat{\Delta}_{w,t} - \mathbb{E}_{S_t}\left[\widehat{\Delta}_{w,t}\right]\right\|^2\right]. \tag{2}$$

Ideally, both bias and variance should be minimized. However, achieving this in practice is challenging due to the uncertain and dynamically changing nature of client availability patterns. Consequently, a bias-variance tradeoff often arises. For instance, FedAvg eliminates bias entirely but incurs high variance, which empirically leads to suboptimal performance in partial participation scenarios. Interestingly, some studies have observed that biased aggregation can, in fact, improve the convergence rate [9], [23]. Based on these observations, we recognize that minimizing bias alone is neither necessary nor always beneficial. Therefore, we shift our focus to reducing, or even minimizing, variance while maintaining bias within an acceptable range.

B. FedALL: Federated Averaging With Most Recent Updates From All Clients

To achieve this goal, we propose FedALL, a novel federated learning algorithm that effectively reduces variance in the aggregation process. Fig. 4 presents an overview of the FedALL algorithm with four clients. In the first round, the server receives updates from clients 1 and 2, caches them, and uses them for aggregation. In the second round, upon receiving updates from client 4, the server aggregates it with the cached updates from clients 1 and 2, then caches client 4's update. In the third round, the server receives updates from clients 3 and 4. It aggregates these with the cached updates from clients 1 and 2, and subsequently updates the cache with the new updates from clients 3 and 4. This process continues until training concludes.

Algorithm 1 FedALL

Server:

1 For the set of clients S, initialize $\mathbf{y}_i$ to 0
2 **for** *t in 1,2,3...,T* **do**
3 Select a subset of clients $S_t \subset S$
4 Send $\mathbf{w}_t$ to S_t
5 Received $\{\Delta^i_{w,t} | i \in S_t\}$ from S_t
6 **if** *enabled storage-efficient module* **then**
7 **for** *each* $\mathbf{y}_i$ **do**
8 **if** *t - round of* $\mathbf{y}_i > r$ **then**
9 Delete $\mathbf{y}_i$
10 **for** $i \in S_t$ **do**
11 $\mathbf{y}_i \leftarrow \Delta^i_{w,t}$
12 Update the round of $\mathbf{y}_i$ to t
13 $\hat{\alpha}_t \leftarrow$ Eq. (4) or Eq. (5) /* FedALL */
14 $\hat{\alpha}_t \leftarrow 0$ /* FedAvg */
15 $\hat{\alpha}_t \leftarrow 1$ /* MIFA */
16 $K_t = M + \hat{\alpha}_t(N - M)$
17 $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \frac{1}{K_t}(\sum_{i \in S_t} \mathbf{y}_i + \sum_{i \notin S_t} \hat{\alpha}_t \mathbf{y}_i)$

Clients:

18 Do client initialization
19 Receive $\mathbf{w}_t$ from server
20 **for** *epoch* τ *in 0,1,2..., E-1* **do**
21 $\mathbf{w}_t^{\tau+1} \leftarrow$ train $(\mathbf{w}_t^{\tau})$
22 $\Delta^i_{w,t} = \mathbf{w}_t^E - \mathbf{w}_t$
23 Send model updates $\Delta^i_{w,t}$ to server

We also provide the detailed workflow of FedALL in Algorithm 1, which includes the storage-efficient module and the computation of the decay coefficient α_t. Upon receiving updates, if enabled, the server first prunes cached updates from clients who have not participated in the last r rounds. This process reduces storage requirements while retaining relatively recent updates that are more relevant for variance reduction. With this module, the server storage requirement drops to $\mathcal{O}(rM)$, making it more practical in real-world scenarios with many clients. However, discarding updates may cause slight accuracy loss compared to original FedALL. We discuss the tradeoff further in Section V.

The coefficient α_t is computed using either (4) or (5). The server then determines the normalization factor K_t and aggregates the updates to compute the new model $\mathbf{w}_{t+1}$. Lines 14 and 15 show that FedAvg and MIFA are special cases of FedALL, where α_t is fixed at 0 and 1, respectively. From this perspective, α_t balances bias and variance. In particular, when $\alpha_t = 0$, bias is minimized, but variance is high. Conversely, when $\alpha_t = 1$, variance is low, but bias is high. On the contrary, FedALL determines α_t dynamically based on clients' updates and participation patterns.

IV. Theoretical Analysis

In this section, we present the theoretical analysis of bias and variance, which is the foundation for understanding

TABLE I
BIAS AND VARIANCE OF AGGREGATION PROCESS

Method	Bias	Variance[1]	Cached updates	Variance reduction
FedAvg	0	$\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t}))$	✗	baseline
MIFA	$\mathcal{O}(\Delta^i_{w,t'_i} - \Delta^i_{w,t})$	$\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t} - \Delta^i_{w,t'_i}))$	✓	No guarantee
FedVARP	0	$\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t} - \Delta^i_{w,t'_i}))$	✓	No guarantee
FedAU	$\mathcal{O}(\Delta^i_{w,t})$	$\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t}))$	✗	No guarantee
SCAFFOLD	0	$\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t}))$	✗	✗
FedAvgAU[2]	$\mathcal{O}(\Delta^i_{w,t})$ 0	$\mathcal{O}(\mu^2\mathrm{TrVar}(\Delta^i_{w,t}))$ [t mod P = 0] $\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t}))$ [t mod P $\neq$ 0]	✗ ✗	✓ (if $0 < \mu < 1$) ✗
FedALL	$\mathcal{O}(\alpha_t(\Delta^i_{w,t'_i} - \Delta^i_{w,t}))$	$\mathcal{O}(\mathrm{TrVar}(\Delta^i_{w,t} - \alpha_t\Delta^i_{w,t'_i}))$	✓	✓

[1] $\Delta^i_{w,t}$ is the update of client i at t-th round; t'_i is client i's most recent active round; $\mathrm{TrVar}(\cdot)$ is the trace of the variance matrix.
[2] μ represents the learning rate and P is a hyperparameter in FedAvgAu.

FedALL. We also analyze the consistency and convergence of FedALL under partial participation.

A. Bias and Variance: Foundation of FedALL

Following the definition in (2), we present the bias and variance of different methods illustrated in Table I, with detailed computations provided in Appendix I. To simplify the notation, we use Var to represent the variance matrix and TrVar to represent the trace of the variance matrix. We use Cov to represent the covariance matrix and TrCov to represent the trace of the covariance matrix.

We begin by analyzing the bias. From Table I, we observe that FedAvg, FedVARP, and SCAFFOLD are unbiased. However, as discussed in Section III, unbiasedness does not necessarily indicate good model performance under partial participation. Empirical studies have shown that FedAvg and SCAFFOLD often exhibit poor performance or even fail to converge under partial participation settings [1], [24], [25], [26]. On the contrary, we argue that reducing variance is critical, as focusing solely on bias does not yield satisfactory results.

We can categorize these methods into two groups based on whether they cached clients' most recent updates. For the methods that do not use clients' most recent updates, the variances of FedAvg, SCAFFOLD, and FedAvgAU are proportional to the variance of $\Delta^i_{w,t}$ across different clients. Here, $\Delta^i_{w,t}$ represents the model update of client i at round t, which is highly correlated with the client's local dataset. As a result, the variance is positively correlated with the heterogeneity of local datasets across clients. The variance of FedAU is much more complex and difficult to simplify, as it assigns a coefficient γ^i_t to each client's update, with these coefficients being determined by the participation history.

The variances of MIFA, FedVARP, and FedALL are proportional to $\mathrm{TrVar}(\Delta^i_{w,t} - \Delta^i_{w,t'_i})$. This term is less influenced by local dataset heterogeneity because both $\Delta^i_{w,t}$ and Δ^i_{w,t'_i} are strongly affected by the local data. By computing their difference, the effects of the local dataset are partially canceled out. Consequently, the variance induced by data heterogeneity is reduced, leading to a smaller overall variance.

Although employing clients' most recent updates to reduce variance seems reasonable, there is no theoretical guarantee. Comparing FedAvg and FedVARP reveal that FedVARP yields lower variance only when $2\mathrm{TrCov}(\Delta^i_{w,t}, \Delta^i_{w,t'_i}) > \mathrm{TrVar}(\Delta^i_{w,t'_i})$. This indicates that variance reduction depends on both the covariance between $\Delta^i_{w,t}$ and Δ^i_{w,t'_i} and the variance of Δ^i_{w,t'_i}. A similar, yet more complex, condition can be derived for MIFA due to the additional term involving the variance of $\Delta^i_{w,t}$.

On the contrary, we can analyze the variance of FedALL from a different perspective. By selecting an appropriate α_t, we can directly minimize the variance. To achieve this, we solve the equation $\mathrm{TrVar}(\Delta^i_{w,t} - \alpha_t\Delta^i_{w,t'_i})$ with respect to α_t

$$\alpha_t = \frac{\mathrm{TrCov}\left(\Delta^i_{w,t}, \Delta^i_{w,t'_i}\right)}{\mathrm{TrVar}\left(\Delta^i_{w,t'_i}\right)}, \quad i \in S. \tag{3}$$

By substituting the above equation into the variance expression for FedALL, we obtain the minimal variance. Comparing this with FedAvg reveals that FedALL achieves lower variance without requiring additional assumptions.

However, obtaining the exact solution is intractable due to partial participation, which makes it impossible to compute the covariance and variance for all clients. To address this, we propose two alternatives. The first option is to use information from the participating clients in each round to compute the variance and covariance

$$\texttt{FedALL-1}: \quad \hat{\alpha}_t = \frac{\mathrm{TrCov}\left(\Delta^i_{w,t}, \Delta^i_{w,t'_i}\right)}{\mathrm{TrVar}\left(\Delta^i_{w,t'_i}\right)}, \quad i \in S_t. \tag{4}$$

A significant limitation of the estimation presented in (4) is that it is strongly influenced by the participation pattern, as it only considers the variance and covariance of active clients. For instance, if the clients participating in round t possess similar data, the variance of Δ^i_{w,t'_i} will decrease, resulting in an inflated value for $\hat{\alpha}_t$ compared to the true value. Conversely, $\hat{\alpha}_t$ might be much smaller than the true value. This inconsistency

suggests that the estimation is unstable across different rounds and may fail to consistently reduce variance. In addition, the time complexity of this estimation is $\mathcal{O}(M|w|)$, where $|w|$ is the size of the model. It is not satisfactory, as it scales with the size of the model.

To address these issues, we propose a lightweight heuristic method for computing α_t. We base this on two observations: first, $\alpha_t = 1$ when $t = t_i'$; second, as $t - t_i'$ approaches infinity, $\mathrm{TrCov}(\Delta_{w,t}^i, \Delta_{w,t_i'}^i)$ should approach zero. This is because model updates are gradient-driven and gradients at increasingly distant time steps become statistically independent. Based on these observations, we make the following assumption: the covariance is proportional to the variance, with the proportionality coefficient being a function of the time gap $t - t_i'$.

First, this assumption is intuitively reasonable. A larger $\mathrm{TrVar}(\Delta_{w,t_i'}^i)$ indicates a more heterogeneous local dataset, which likely leads to a larger $\mathrm{TrCov}(\Delta_{w,t}^i, \Delta_{w,t_i'}^i)$, since the covariance is also largely influenced by the local dataset. Second, from a theoretical perspective, we can decompose $\Delta_{w,t}^i$ into $D_i + \xi_{i,t} + w_{t-1}$, where D_i represents the contribution from the local dataset, $\xi_{i,t}$ is the noise term, and w_{t-1} is the global model from the previous round. We can rewrite the covariance and variance as follows:

$$\begin{aligned}
&\mathrm{Cov}\left(\Delta_{w,t}^i, \Delta_{w,t_i'}^i\right) \\
&\quad = \mathrm{Var}(D_i) + \mathrm{Cov}\left(D_i, w_{t-1}\right) \\
&\qquad + \mathrm{Cov}\left(D_i, w_{t_i'-1}\right) + \mathrm{Cov}\left(w_{t-1}, w_{t_i'-1}\right) \\
&\mathrm{Var}\left(\Delta_{w,t_i'}^i\right) \\
&\quad = \mathrm{Var}(D_i) + \mathrm{Var}\left(\xi_{i,t_i'}\right) + \mathrm{Var}\left(w_{t_i'-1}\right) \\
&\qquad + 2\mathrm{Cov}\left(D_i, w_{t_i'-1}\right)
\end{aligned}$$

where we assume that the noise term $\xi_{i,t}$ is independent of the local dataset contribution D_i and the global model w_{t-1}.

Next, we analyze the difference between the covariance and variance by each term. First, the $\mathrm{Var}(D_i)$ and $\mathrm{Cov}(D_i, w_{t_i'-1})$ terms appear in both expressions. Second, the $\mathrm{Var}(\xi_{i,t_i'})$ term can be treated as a constant term since it is independent of the local dataset contribution and the global model. Therefore, the major difference lies between the $\mathrm{Cov}(D_i, w_{t-1}) + \mathrm{Cov}(w_{t-1}, w_{t_i'-1})$ term in the covariance and the $\mathrm{Var}(w_{t_i'-1}) + Cov(D_i, w_{t_i'-1})$ term in the variance. Regarding $\mathrm{Cov}(D_i, w_{t-1})$ and $\mathrm{Cov}(D_i, w_{t_i'-1})$, since the global model aggregates updates from all clients, the covariance between the local dataset contribution and the global model should remain relatively stable across different time steps.

Both $\mathrm{Cov}(w_{t-1}, w_{t_i'-1})$ and $\mathrm{Var}(w_{t_i'-1})$ depend on the global model at different time steps. As the time gap $t - t_i'$ increases, the covariance between the global models at these two time steps should decrease, while the variance of the global model at time step $t_i' - 1$ should remain relatively stable. Therefore, it is reasonable to assume that the covariance is proportional to the variance, with the proportionality coefficient being a function of the time gap $t - t_i'$.

Based on the above observations and assumptions, we choose a simple form of equation to describe such proportionality

$$\texttt{FedALL-2}\colon\ \hat{\alpha}_t = \frac{1}{\mathbb{E}\left[t - t_i'\right]^{\lambda}}. \tag{5}$$

The optimal coefficient α_t in (3) follows from the variance-minimization analysis, but this derivation holds only in an idealized setting. Under partial client participation, exact computation becomes intractable. Consequently, (5) serves as a practical surrogate, motivated by the qualitative expectation that cross-round covariance diminishes as the temporal gap $t - t_i'$ increases. Among plausible monotone decay rules, we select this specific functional form primarily as a lightweight design choice. It only introduces a single hyperparameter λ to govern the decay rate, avoiding the complexity of hyperparameter tuning associated with more complex models.

We emphasize that the chosen decay form is one reasonable approximation, rather than a formally derived expression. This design incurs negligible computational overhead, requiring only scalar operations independent of model scale. It demonstrates robust performance with a default setting of $\lambda = 1$ across diverse benchmarks. We acknowledge the heuristic nature of this choice as a limitation of our current framework. Future work may explore more expressive functional forms or data-driven estimators to refine the underlying covariance–variance dynamics.

As mentioned earlier, FedAvg and MIFA can be viewed as special cases of FedALL where α_t is fixed at 0 and 1, respectively. From this perspective, λ acts as a hyperparameter controlling the bias-variance trade-off. FedAvg corresponds to $\lambda = \infty$, implying that past updates are completely uncorrelated with current updates. MIFA corresponds to $\lambda = 0$, indicating that past updates are fully correlated with current updates.

B. Consistency

Consistency is another crucial property of an estimator, defined as the convergence of the estimator to the true value as the number of samples increases. In the context of partial participation, we define the consistency of the aggregation process as the convergence of the aggregation result under partial participation to that under full participation as the number of active clients grows. We prove that FedALL is consistent in the following theorem.

Theorem 1 (Consistency): Given the set of active clients in round t as S_t and the set of all clients S, FedALL satisfies

$$\lim_{S_t \to S} \widehat{\Delta}_{w,t} = \Delta_{w,t}.$$

Proof: By definition from Algorithm 1, we know that

$$\widehat{\Delta}_{w,t} = \frac{1}{K_t}\left(\sum_{i \in S_t} \Delta_{w,t}^i + \sum_{i \notin S_t} \hat{\alpha}_t \Delta_{w,t_i'}^i\right).$$

When $S_t = S$ (i.e., $M = N$), we can simplify the above equation as follows:

$$\widehat{\Delta}_{w,t} = \frac{1}{N}\sum_{i \in S} \Delta_{w,t}^i = \Delta_{w,t} \quad \square.$$

C. Convergence

We provide the convergence analysis of `FedALL` under partial participation in a nonconvex setting. We begin with the following three standard assumptions:

Assumption 1 (L-Smoothness): Each client's loss function $f_i(\cdot)$ is L-smooth, i.e., $\|\nabla f_i(x) - f_i(y)\| \le L\|x - y\|$, for all $i \in [N]$.

Assumption 2 [Unbiased Gradient Estimator and Bounded Local Variance]: The stochastic gradient of each client is an unbiased estimator of the gradient, i.e., $\mathbb{E}[\nabla f_i(w;\xi)] = \nabla f_i(w)$, where ξ represents the randomness of gradients. The variance of each client's stochastic gradients is bounded by σ_l^2, i.e., $\mathbb{E}\|\nabla f_i(w;\xi_i) - f_i(w)\|^2 \le \sigma_l^2$, for all $i \in [N]$.

Assumption 3 (Bounded Global Variance): The variance of each client's gradients is bounded by σ_g^2, i.e., $\mathbb{E}\|\nabla f_i(w) - f(w)\|^2 \le \sigma_g^2$, for all $i \in [N]$.

These assumptions are all basic assumptions in the convergence analysis of federated learning algorithms [8], [16], [27], [28]. Notably, we do not require assumptions such as the bounded gradients assumption or Hessian Lipschitz [6], [17], [29]. The proof is deferred to Appendix II.

Theorem 2 (Convergence): Given the local training epochs E, the number of communication rounds T, the total number of clients N, the number of active clients M, the local learning rate η, Assumptions 1–3, and for η and α_t such that $\eta \le \min\{(\sqrt{3}/24EL), (M/75ELN)\}$ and $\alpha_t \le (\sqrt{2}/12(N-M))$, FedALL satisfies

$$\min_{t\in[0,T]} \mathbb{E}\|\nabla f(\mathbf{w}_t)\|^2 \le \frac{4\mathbb{E}\left[f(\mathbf{w}_0) - f(\mathbf{w}_T)\right]}{\eta E T} + \frac{4}{\eta E}\Phi_1\sigma_l^2 + \frac{4}{\eta E}\Phi_2\sigma_g^2.$$

Here, Φ_1 and Φ_2 are defined as follows:

$$\Phi_1 = \frac{2\eta^3 E(E-1)L^2}{K_t^2}(M^2 + 9\alpha_t^2(N-M)^2) + \frac{\eta^2 E^2 L^2 N}{K_t}\left(6\eta^2(E-1)L^2 + 1\right)$$

$$\Phi_2 = \frac{4\eta E}{K_t^2}\left(M^2 D + \alpha_t^2(N-M)^2(3+3D)\right) + \frac{\eta^2 E^2 L N}{K_t}(12D+3); \quad D = 2\eta^2(E-1)L^2.$$

Corollary 1: For a large T, by setting $\eta = \sqrt{(M/ET)}$ and $\alpha_t = (M/(N-M)\sqrt{NT((M/24)+(73/8N))})$, FedALL converges to a stationary point with the following convergence rate:

$$\min_{t\in[0,T]} \mathbb{E}\|\nabla f(\mathbf{w}_t)\|^2 \le \mathcal{O}\left(\frac{\mathbb{E}\left[f(\mathbf{w}_0) - f(\mathbf{w}_T)\right]}{\sqrt{EMT}}\right) + \mathcal{O}\left(\frac{\sigma_l^2\sqrt{EN}}{\sqrt{MT}}\right) + \mathcal{O}\left(\frac{\sigma_g^2\sqrt{EN}}{\sqrt{MT}}\right).$$

Through convergence analysis, we demonstrate that FedALL converges at a rate of $\mathcal{O}(1/\sqrt{T})$, which is equivalent to the rates of FedAvg, MIFA, and FedVARP. In addition, we find that the convergence proof for MIFA fails without additional assumptions, such as Hessian Lipschitzness. This issue arises because MIFA sets $\alpha_t = 1$, which does not satisfy our derived condition $\alpha_t \le (\sqrt{2}/12(N-M))$. On the contrary, FedAvg fulfills this condition by setting $\alpha_t = 0$, allowing its proof to proceed without requiring further assumptions.

V. Experiments

In this section, we present the experimental results of our proposed algorithm. We compare FedALL with various state-of-the-art federated learning algorithms that focus on addressing the partial participation problem or reducing variance. In addition, we evaluated two storage-efficient variants of FedALL: FedALL-2-r2 and FedALL-2-r5. FedALL-2-r2 caches clients' updates for two rounds, while FedALL-2-r5 caches clients' updates for five rounds. We evaluated all methods on different datasets and at various levels of client participation and data heterogeneity to comprehensively assess the performance of each method.

A. Experimental Setup

We evaluated all methods on four datasets: EMNIST, CIFAR-10, CIFAR-100, and HAR [30], [31], [32]. EMNIST, CIFAR-10, and CIFAR-100 are classic image classification datasets, while HAR is a human activity recognition dataset that records user activities using smartphones. For FedALL, we evaluated both approaches: FedALL-1, which computes α_t using (4), and FedALL-2, which computes α_t using (5) with $\lambda = 1$. Each task was executed five times with different random seeds.

1) Accuracy and Standard Deviation: We report both the average *accuracy (Acc)* and the average *standard deviation (Std Dev)* to evaluate the model performance. Accuracy and standard deviation are standard metrics in the model performance evaluation. Furthermore, in the context of partial participation, the standard deviation can serve as an implicit indicator of the variance in model aggregation. A high variance in model aggregation typically results in significant differences in the aggregated model for different random seeds, potentially leading to a larger standard deviation in the accuracy.

Considering both accuracy and standard deviation, we also calculated the *performance score (Perf-S)* for all methods, defined as the sum of the *accuracy score (Acc-S)* and the *standard deviation score (Std Dev-S)*. The definitions of these scores are as follows:

$$\begin{aligned}\text{Perf-S (method A)} &= \text{Acc-S (A)} + \text{Std Dev-S (A)}\\ \text{Acc-s (A)} &= \text{Acc (A)} - \text{Acc (FedAvg)}\\ \text{Std Dev-S (A)} &= \text{Std Dev (FedAvg)} - \text{Std Dev (A)}.\end{aligned} \quad (6)$$

The rationale behind these definitions is that a higher accuracy and a smaller standard deviation compared to FedAvg indicate better performance. The performance score considers both accuracy and standard deviation, enabling a comprehensive evaluation of each method's performance.

2) Models and Training Settings: The EMNIST dataset contains 112 800 training samples and 18 800 testing samples across 47 classes. The CIFAR-10 dataset consists of 50 000 training samples and 10 000 testing samples in ten classes. The CIFAR-100 dataset contains 50 000 training samples and 10 000 testing samples in 100 classes. The HAR dataset includes 7352 training samples from 21 users and 2947 testing samples from nine users across six classes. For the EMNIST dataset, we trained LeNet-5 models for 5000 rounds [33]. On CIFAR-10 and CIFAR-100, we trained ResNet-18 models for 2000 and 3000 rounds, respectively [34]. We set the batch size to 64, the number of local epochs to one, and the optimizer to SGD for all experiments. To ensure appropriate learning rates for different algorithms on

different tasks, we used the training loss as an indicator and tuned learning rates for each task over a grid of values: $\{10^{-2}, 10^{-1.75}, 10^{-1.5}, 10^{-1.25}, 10^{-1}, 10^{-0.75}, 10^{-0.5}\}$ for all methods except FedAvgAU. For FedAvgAU, we multiplied each learning rate in the grid by 0.01 to prevent loss explosion [15]. We applied the learning rate that yielded the smallest average training loss, computed over the last 20 rounds at 2500, 1000, and 1500 rounds for EMNIST, CIFAR-10, and CIFAR-100, respectively. For the HAR dataset, we used LSTM models with the default Adam optimizer for 600 rounds [35], [36]. The final test set accuracy and its standard deviation were averaged over the last 20 training rounds.

3) Clients and Data Distribution: To comprehensively evaluate different methods under the partial participation setting, we considered various levels of client participation and data heterogeneity. We established four scenarios for client participation, with a total number of 50, 100, 200, and 500 clients participating in training, respectively. In each round, ten clients were randomly sampled from the total pool, resulting in participation ratios of 20%, 10%, 5%, and 2%. In this article, we denote the sampling of A clients from a total of B clients as $A - B$. For example, 10–50 indicates a sampling of ten clients from a total of 50 clients. As the participation ratio decreases, the task becomes increasingly challenging. For the HAR dataset, the local datasets were naturally partitioned by users. We randomly sampled three and seven clients from the 21 users in each round, resulting in participation ratios of 14.3% and 33.3%, respectively.

Regarding data heterogeneity, we employed the Dirichlet distribution to sample local datasets for each client [26], [37]. The concentration parameter β of the Dirichlet distribution controls the level of data heterogeneity. We set $\beta = 0.5$ and $\beta = 0.1$ to represent low-data and high-data heterogeneity, respectively. A smaller β indicates a more heterogeneous data distribution across different clients, making the task more challenging.

B. Main Results on Computer Vision Datasets

We present the experimental results in two tables. The accuracies and standard deviations of different methods for each task are illsutrated in Table II. The average accuracy scores, standard deviation scores, and performance scores for each method are summarized in Table III.

1) FedALL: Both FedALL-1 and FedALL-2 exhibit good performance in terms of average accuracy and standard deviation. Notably, FedALL-2 outperforms all other methods on both accuracy and standard deviation. The difference between these two methods lies in their calculation of α_t. In FedALL-2, α_t is heuristically computed by using the most recent active rounds of all clients, whereas in FedALL-1, α_t relies on the variance and covariance of clients participating in the current round. Empirical results suggest that the heuristic computation α_t in FedALL-2 is more effective. One possible explanation is that α_t in FedALL-2 better captures the participation patterns of all clients, making it more robust and advantageous for aggregation compared to FedALL-1.

Regarding variance reduction, while standard deviations can implicitly indicate variance, it is more convincing to compute and compare the variance of aggregation directly. Fortunately, we can conduct experiments under special settings, which are impractical in real-world scenarios, to further verify the effects on variance reduction. In these experiments, N clients participated in each round, but only the updates from M clients were used for aggregation. Since we had the information of all N clients, we were able to calculate the variance by definition (2). Meanwhile, this setting preserved the partial participation pattern, as only M clients were used for aggregation in each round. The variances are presented in Fig. 5. We conducted experiments on EMNIST, CIFAR-10, and CIFAR-100 using $Dir(0.5)$, $Dir(0.1)$ distributions, along with the 10–100 participation pattern. In addition to the baseline FedAvg, we also included MIFA and FedVARP for comparison.

The results demonstrate that MIFA and FedALL-2 are the most effective methods for variance reduction, while FedVARP achieves only a slight reduction. The empirical variances of FedVARP and MIFA align with our previous theoretical analysis, which indicates that the variance of MIFA is smaller than that of FedVARP. Comparing the theoretical variance of FedALL and MIFA, the coefficient of FedALL is N^2/K_t^2 times larger than that of MIFA, as $M < K_t < N$. However, we empirically find that the variance of FedALL-2 is roughly equivalent to that of MIFA across all experiments. This observation suggests that $\mathrm{tr}(\mathrm{Var}(\Delta^i_{w,t} - \alpha_t \Delta^i_{w,t'_i}))$ should be approximately K_t^2/N^2 smaller than $\mathrm{tr}(\mathrm{Var}(\Delta^i_{w,t} - \bar{\Delta}^i_{w,t}))$. This supports our expectation that the term $\mathrm{tr}(\mathrm{Var}(\Delta^i_{w,t} - \bar{\Delta}^i_{w,t}))$ can be reduced by incorporating the coefficient α_t. From these results, we conclude that MIFA, FedVARP, and FedALL-2 achieve different degrees of variance reduction, with FedALL-2 and MIFA significantly outperforming FedVARP and FedAvg. These conclusion is consistent with those derived from standard deviation scores, further validating the use of standard deviation as an indicator for variance reduction.

2) Storage-Efficient FedALL: We further propose two storage-efficient variants of FedALL-2, denoted as FedALL-2-r2 and FedALL-2-r5, which retain only the most recent model updates from active clients within the last two and five rounds, respectively. These variants reduce the server storage cost from $\mathcal{O}(N|w|)$ to $\mathcal{O}(2M|w|)$ and $\mathcal{O}(5M|w|)$. More generally, when caching updates from the last p rounds, the storage savings are $\mathcal{O}(|w|(N - pM))$, which becomes significant for large model size and large client population.

Although FedALL-2-r2 and FedALL-2-r5 achieve slightly lower performance than the full FedALL-2, ranking second and fourth, they remain competitive with other state-of-the-art methods. This mild degradation is expected due to reduced historical information. Nevertheless, both variants consistently outperform FedAvg across most tasks, demonstrating that they preserve the core benefits of FedALL-2 while offering practical memory efficiency.

3) MIFA and FedVARP: In Section IV, we theoretically proved that the effects of MIFA and FedVARP on variance reduction are not guaranteed. Experimental results indicate that the Std Dev scores of FedVARP and MIFA can be either negative or positive depending on experimental settings, reflecting unstable effects on variance reduction. On average across all tasks, MIFA reduces the standard deviation by 0.65%, while FedVARP achieves a reduction of

TABLE II

ACCURACY AND STANDARD DEVIATION ON TEST SET. FOR EACH TASK, THE BEST RESULT IS HIGHLIGHTED IN BOLD. SCORES OF INDIVIDUAL TASKS ARE DEFINED IN (6). EACH SCORE IS THE SUMMATION OF CORRESPONDING SCORES OF EACH TASK IN A ROW.

Dataset	*Dir*	Method	10-50	10-100	10-200	10-500	Acc-S	Std Dev-S	Perf-S
EMNIST	0.5	FedAvg	82.49 ±0.50	82.15 ±0.53	81.99 ±0.60	81.11 ±0.82	0.00	0.00	0.00
		MIFA	83.51 ±0.25	82.81 ±0.32	82.37 ±0.57	81.00 ±0.27	1.95	1.04	2.99
		FedVARP	83.58 ±0.28	82.83 ±0.45	81.92 ±0.75	80.69 ±0.78	1.28	0.19	1.47
		FedAU	82.83 ±0.54	82.36 ±0.50	82.02 ±0.50	80.77 ±0.95	0.24	-0.04	0.20
		SCAFFOLD	**83.99** ±0.27	82.82 ±0.35	81.75 ±0.42	79.45 ±0.40	0.27	1.01	1.28
		FedAvgAU	78.79 ±0.63	77.49 ±0.42	75.62 ±0.55	68.90 ±0.67	-26.94	0.18	-26.76
		FedALL-1	83.71 ±0.22	83.23 ±0.24	82.80 ±0.31	81.56 ±0.37	3.56	**1.31**	4.87
		FedALL-2	83.68 ±0.30	**83.48** ±0.39	**82.82** ±0.37	**82.02** ±0.37	**4.26**	1.02	**5.28**
		FedALL-2-r2	82.61 ±0.44	82.54 ±0.45	82.29 ±0.50	80.99 ±0.95	0.69	0.11	0.80
		FedALL-2-r5	83.01 ±0.40	82.66 ±0.54	82.37 ±0.43	81.23 ±0.66	1.53	0.42	1.95
	0.1	FedAvg	80.10 ±0.66	80.01 ±0.70	79.61 ±0.88	79.01 ±1.01	0.00	0.00	0.00
		MIFA	81.96 ±0.31	81.29 ±0.50	80.44 ±0.83	78.77 ±0.45	3.73	1.16	4.89
		FedVARP	81.95 ±0.45	81.76 ±0.37	80.99 ±0.65	79.81 ±0.68	5.78	1.10	6.88
		FedAU	80.36 ±0.74	80.11 ±0.68	79.53 ±0.77	79.43 ±0.66	0.70	0.40	1.10
		SCAFFOLD	78.58 ±0.96	78.48 ±0.38	76.23 ±0.59	73.79 ±1.74	-11.65	-0.42	-12.07
		FedAvgAU	73.49 ±1.17	73.20 ±0.90	69.38 ±0.75	65.31 ±0.63	-37.35	-0.20	-37.55
		FedALL-1	**82.10** ±0.23	**82.12** ±0.33	80.98 ±0.33	80.33 ±0.25	6.80	**2.11**	**8.91**
		FedALL-2	82.04 ±0.52	81.60 ±0.50	**81.53** ±0.51	**80.97** ±0.44	**7.41**	1.28	8.69
		FedALL-2-r2	81.11 ±0.53	80.05 ±0.60	79.98 ±0.68	80.02 ±0.86	2.43	0.58	3.01
		FedALL-2-r5	81.23 ±0.45	81.23 ±0.43	80.34 ±0.53	80.09 ±0.71	4.16	1.06	5.22
CIFAR-10	0.5	FedAvg	91.16 ±0.65	88.31 ±1.02	86.52 ±1.90	82.12 ±2.41	0.00	0.00	0.00
		MIFA	91.15 ±0.40	89.25 ±0.35	85.82 ±0.94	79.12 ±1.14	-2.77	3.15	0.38
		FedVARP	90.70 ±1.09	89.30 ±1.38	85.92 ±2.18	79.41 ±2.93	-2.78	-1.60	-4.38
		FedAU	90.62 ±1.27	90.66 ±0.97	87.69 ±2.13	83.06 ±2.81	3.92	-1.20	2.72
		SCAFFOLD	90.82 ±0.24	89.98 ±0.42	87.20 ±1.49	81.41 ±2.39	1.30	1.44	2.74
		FedAvgAU	86.31 ±0.94	84.73 ±0.89	81.80 ±1.07	74.89 ±0.83	-20.38	2.25	-18.13
		FedALL-1	91.12 ±0.49	89.59 ±0.52	85.79 ±1.95	78.13 ±4.23	-3.48	-1.21	-4.69
		FedALL-2	**91.68** ±0.28	**90.86** ±0.39	**88.86** ±0.82	**84.88** ±1.24	**8.17**	**3.25**	**11.42**
		FedALL-2-r2	91.13 ±0.48	89.43 ±0.83	87.08 ±1.35	83.02 ±1.72	2.55	1.60	4.15
		FedALL-2-r5	91.27 ±0.31	89.71 ±0.59	87.49 ±1.00	83.51 ±1.36	3.87	2.72	6.59
	0.1	FedAvg	77.42 ±3.43	68.90 ±5.07	68.49 ±4.66	59.01 ±4.33	0.00	0.00	0.00
		MIFA	**82.63** ±1.00	69.61 ±5.58	73.94 ±2.02	57.71 ±4.64	10.07	3.25	13.32
		FedVARP	80.56 ±1.89	77.62 ±3.44	70.12 ±4.64	58.94 ±4.07	13.42	2.45	15.87
		FedAU	77.56 ±4.24	70.82 ±6.47	65.55 ±5.13	58.91 ±5.12	-0.98	-4.47	-5.45
		SCAFFOLD	73.66 ±4.95	45.91 ±3.51	35.69 ±6.52	49.78 ±4.92	-68.78	-3.41	-72.19
		FedAvgAU	70.50 ±3.07	64.76 ±3.82	58.30 ±4.34	48.81 ±1.19	-31.45	4.07	-27.38
		FedALL-1	81.64 ±1.42	**79.18** ±1.93	71.04 ±2.90	54.80 ±7.34	12.84	2.90	15.74
		FedALL-2	81.32 ±2.18	78.77 ±2.83	**74.00** ±3.62	**67.35** ±2.62	**27.62**	**5.24**	**32.86**
		FedALL-2-r2	79.43 ±2.52	72.10 ±3.87	70.29 ±3.02	61.75 ±3.54	9.75	3.54	13.29
		FedALL-2-r5	79.25 ±2.22	73.83 ±3.51	71.18 ±3.77	62.40 ±2.93	12.84	4.06	16.90
CIFAR-100	0.5	FedAvg	68.23 ±0.89	66.99 ±1.21	66.55 ±1.62	66.05 ±1.84	0.00	0.00	0.00
		MIFA	70.59 ±0.43	67.60 ±0.79	63.95 ±0.83	61.35 ±0.70	-4.33	2.81	-1.52
		FedVARP	69.99 ±0.81	67.88 ±1.55	65.77 ±1.99	64.32 ±2.08	0.14	-0.87	-0.73
		FedAU	69.64 ±0.81	69.48 ±2.13	67.66 ±2.07	67.36 ±1.91	6.32	-1.36	4.96
		SCAFFOLD	69.23 ±1.49	69.73 ±1.27	**69.50** ±0.88	67.03 ±0.82	7.67	1.10	8.77
		FedAvgAU	64.76 ±0.37	63.01 ±1.00	58.90 ±0.70	49.06 ±0.53	-32.09	2.96	-29.13
		FedALL-1	70.24 ±0.35	68.83 ±0.86	66.15 ±0.67	64.49 ±0.60	1.89	**3.08**	4.97
		FedALL-2	**70.77** ±0.40	**70.18** ±0.74	68.88 ±0.84	**67.78** ±0.75	**9.79**	2.83	**12.62**
		FedALL-2-r2	70.22 ±0.50	66.57 ±0.78	68.11 ±0.95	65.58 ±0.91	2.66	2.42	5.08
		FedALL-2-r5	70.43 ±0.47	67.43 ±1.02	68.21 ±0.88	65.99 ±0.88	4.24	2.31	6.55
	0.1	FedAvg	62.49 ±1.87	64.92 ±1.92	64.11 ±2.55	62.01 ±2.52	0.00	0.00	0.00
		MIFA	65.13 ±0.89	64.57 ±1.20	60.60 ±0.86	57.10 ±1.65	-6.13	4.26	-1.87
		FedVARP	64.85 ±1.30	66.24 ±1.21	62.20 ±2.40	61.54 ±2.10	1.30	1.85	3.15
		FedAU	63.19 ±2.20	64.51 ±1.47	64.16 ±2.93	62.35 ±1.96	0.68	0.30	0.98
		SCAFFOLD	63.39 ±0.92	67.45 ±0.67	66.52 ±0.84	**65.15** ±0.98	8.98	5.45	14.43
		FedAvgAU	57.48 ±0.91	58.07 ±0.65	53.93 ±0.72	44.36 ±1.12	-39.69	**5.46**	-34.23
		FedALL-1	64.91 ±0.74	65.50 ±0.86	62.39 ±1.04	59.08 ±1.28	-1.65	4.94	3.29
		FedALL-2	**66.50** ±1.34	**67.47** ±0.62	**66.56** ±0.92	64.25 ±1.07	**11.25**	4.91	**16.61**
		FedALL-2-r2	63.04 ±1.51	64.88 ±1.28	64.01 ±1.75	62.35 ±1.48	0.75	2.84	3.59
		FedALL-2-r5	64.09 ±1.09	65.12 ±1.29	63.98 ±1.18	62.59 ±1.09	1.85	4.21	6.06

only 0.13% compared to FedAvg. This suggests that MIFA is more effective at reducing variance, consistent with our theoretical analysis. Regarding accuracy, MIFA shows an improvement of only 0.11%, whereas FedVARP demonstrates a much better accuracy improvement of 0.80%. This difference can be attributed to the significant bias associated with MIFA during aggregation, while FedVARP is unbiased. In conclusion, both FedVARP and MIFA outperform FedAvg, with FedVARP being slightly better than MIFA in terms of accuracy.

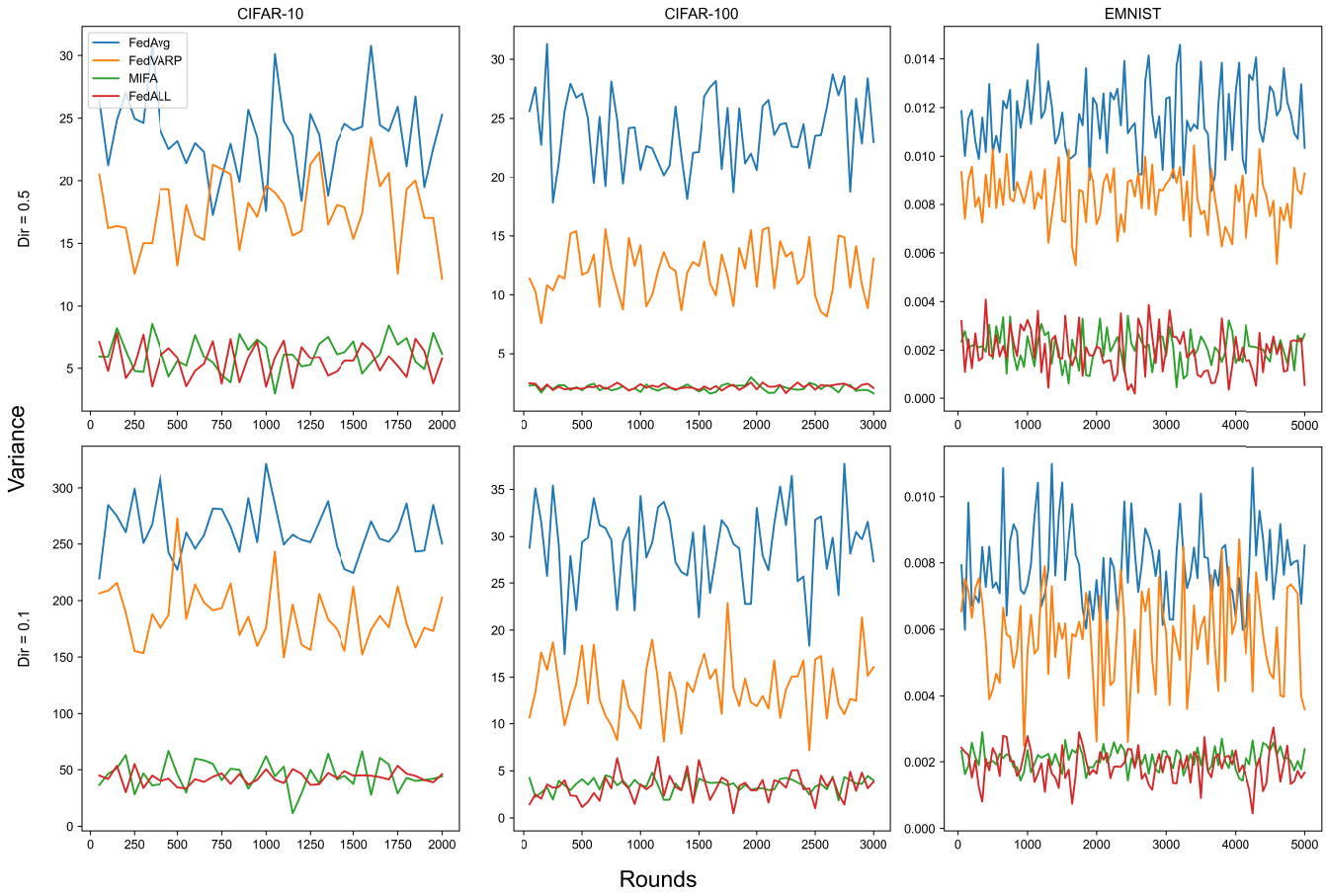

Fig. 5. Variance of aggregated models for different methods under CIFAR-10, CIFAR-100, and EMNIST with the 10–100 participation pattern. The x-axis represents the training rounds, and the y-axis represents the variance of the aggregated models.

TABLE III

AVERAGE PERFORMANCE OF EACH METHOD OVER 24 DIFFERENT TASKS. SCORES ARE DEFINED IN (6). THE "> FEDAVG" IS THE NUMBER OF TASKS, WHERE THE METHOD HAS HIGHER ACCURACIES OR SMALLER STANDARD DEVIATIONS THAN THOSE OF FEDAVG. FOR EXAMPLE, "24–24" MEANS THE METHOD HAS HIGHER ACCURACIES IN ALL 24 TASKS AND SMALLER STANDARD DEVIATIONS IN 24 TASKS COMPARED TO FEDAVG. THE BEST RESULT FOR EACH METRIC IS HIGHLIGHTED IN BOLD

Methods	Acc-S	Std Dev-S	Perf-S	> FedAvg
FedAvg	0.00	0.00	0.00	0-0
MIFA	0.11	0.65	0.76	13-22
FedVARP	0.80	0.13	0.93	14-15
FedAU	0.45	-0.27	0.18	18-9
SCAFFOLD	-2.60	0.22	-2.38	12-17
FedAvgAU	-7.83	0.61	-7.22	0-20
FedALL-1	0.83	0.55	1.38	16-21
FedALL-2	**2.85**	**0.77**	**3.62**	**24-24**
FedALL-2-r2	0.78	0.46	1.24	18-22
FedALL-2-r5	1.19	0.62	1.81	22-23

4) SCAFFOLD: SCAFFOLD exhibits poor average performance in terms of accuracy, with a decrease of 2.60% compared to FedAvg. Notably, SCAFFOLD performs worst on CIFAR-10 with *Dir*(0.1), which aligns with observations from previous research [26]. We also observe that SCAFFOLD performs poorly on CIFAR-10-*Dir*(0.1) and EMNIST-10-*Dir*(0.1), but performs relatively better on CIFAR100-100-*Dir*(0.1). This suggests that SCAFFOLD lacks robustness across different scenarios, particularly when the data distribution is highly heterogeneous. Such a limitation restricts its applicability in real-world scenarios characterized by highly heterogeneous data. Conversely, the standard deviation scores of SCAFFOLD are positive, indicating that it effectively reduces variance. This reduction can be attributed to its use of control variates to mitigate the variance caused by data heterogeneity.

5) FedAvgAU: This method performs poorly across all tasks, with an average decrease of 7.83% in accuracy. This decline is primarily due to its requirement for a much smaller learning rate; otherwise, the loss will explode during the amplification stage. The smaller learning rate significantly slows convergence, requiring over 300 000 training rounds to converge on FashionMNIST and CIFAR-10 [15]. Given this, it is expected to experience substantial accuracy loss when trained with far fewer than 300 000 training rounds, while all other methods are able to converge within a reasonable number of rounds. Such an extremely high number of training rounds is also impractical and inefficient in real-world scenarios. On the contrary, the small learning rates of FedAvgAU yield good standard deviation scores. Theoretically, the variance is positively proportional to the learning rate because $\mathrm{Var}(\Delta^i_{w,t}) = \mathrm{Var}\left(-\eta \sum_{j=0}^{E-1} \nabla f_i\right) \propto \eta^2$, where ∇f_i represents the gradient. By employing a smaller learning rate, this method

TABLE IV
PERFORMANCE OF DIFFERENT METHODS ON THE HAR DATASET. THE BEST RESULT FOR EACH METRIC IS HIGHLIGHTED IN BOLD

Methods	7-21			3-21			Acc-S	Std Dev-S	Perf-S
	E=1	E=5	E=10	E=1	E=5	E=10			
FedAvg	89.23 ±1.20	88.18 ±1.15	85.32 ±1.52	88.25 ±1.80	85.71 ±2.51	83.23 ±2.29	0.00	0.00	0.00
MIFA	89.57 ±0.90	**89.21 ±0.83**	83.13 ±1.31	87.90 ±1.99	87.25 ±1.18	81.72 ±4.43	-1.14	-1.07	-1.31
FedVARP	89.46 ±0.61	88.40 ±0.95	86.05 ±1.44	88.85 ±1.13	85.80 ±2.31	81.49 ±3.07	0.13	0.96	1.09
FedAU	89.67 ±1.30	87.76 ±1.41	85.45 ±1.21	88.57 ±1.95	87.52 ±2.04	83.59 ±3.15	2.64	-0.59	2.05
SCAFFOLD	84.40 ±9.34	41.92 ±9.97	28.90 ±7.39	23.04 ±6.21	fail	fail	-341.66	-22.44	-364.10
FedALL-1	89.69 ±0.95	87.95 ±1.66	**86.78** ±1.47	89.30 ±0.94	87.26 ±0.94	84.01 ±2.04	5.07	2.47	7.54
FedALL-2	90.02 ±0.94	88.12 ±0.58	86.60 ±1.16	**89.55 ±0.88**	**87.63 ±1.26**	**84.77 ±2.01**	**6.67**	**3.64**	**10.68**
FedALL-2-r2	89.66 ±0.80	88.09 ±0.91	85.62 ±1.52	89.07 ±0.84	85.98 ±1.69	82.33 ±2.25	0.83	2.64	3.29
FedALL-2-r5	**90.08 ±0.53**	88.07 ±1.66	85.55 ±1.55	88.92 ±1.17	85.95 ±1.45	83.66 ±1.91	2.31	2.20	4.79

effectively reduces variance at the cost of significant accuracy loss.

6) FedAU: FedAU improves the average accuracy by 0.45%, but increases the average standard deviation by 0.27% compared to FedAvg. The increase in average accuracy is expected because FedAU is specifically designed to address the partial participation problem [17]. However, it has a negative impact on the standard deviation, which is not surprising given that FedAU lacks a potential theoretical foundation for variance reduction. This observation aligns with our previous theoretical analysis, which indicates that the variance of FedAU largely depends on the clients' participating history.

C. Main Results on HAR Dataset

We present the experimental results on the HAR dataset summarized in Table IV. We excluded FedAvgAU from these experiments because it performs poorly on computer vision datasets, requiring an extremely small learning rate and a very large number of training rounds to converge.

The results on the HAR dataset are consistent with those on the computer vision datasets: FedALL-2 achieves the best performance in terms of both accuracy and standard deviation, while FedALL-1 ranks second. In particular, FedALL-2 improves average accuracy by 1.11% and reduces standard deviation by 0.61% compared to FedAvg. Overall, these improvements are smaller on the HAR dataset than on the computer vision datasets. Notably, FedAU becomes the third-best method, outperforming FedVARP and MIFA. These findings suggest that while the performance of these methods depends on specific dataset characteristics, FedALL-2 remains robust across different scenarios and achieves the best performance.

One notable observation is that FedALL-2 fails to outperform the baseline FedAvg in the specific case of the 7–21 participation pattern with five local epochs, showing a tiny performance drop of 0.06% in accuracy. This is the only instance among all tasks where FedALL-2 underperforms FedAvg. In this setting, MIFA is the best method with an accuracy improvement of 1.03%. Since MIFA is a special case of FedALL-2 where $\lambda = 0$, this implies that while the default $\lambda = 1$ is a good starting point, it is not guaranteed to be optimal for all settings. Since there is no strict theoretical guidance for tuning λ, empirical tuning could be considered a limitation for FedALL-2. We will discuss the effects of hyperparameter tuning in Section V. Nevertheless, FedALL-2 still achieves the best average performance across all tasks, demonstrating its robustness and effectiveness in various scenarios.

TABLE V
EFFECTS OF VARIANCE REDUCTION OF FEDALL

Dataset	Model	Std Dev Reduction
EMNIST	LeNet-5	0.38
CIFAR-10	ResNet-18	1.42
CIFAR-100	ResNet-18	1.29
CIFAR-10	MobileNetV2	0.78
CIFAR-10	EfficientNet-B0	2.38
HAR	LSTM	0.61

Another observation is that SCAFFOLD performs poorly on the HAR dataset. Except for the 7–21 participation pattern with one local epoch, where it achieves 84.40% accuracy, SCAFFOLD yields extremely low accuracies on all other tasks and even fails to converge on the 3–21 participation pattern with 5 and ten local epochs. This aligns with the results on the computer vision datasets, further indicating that SCAFFOLD lacks robustness across different scenarios.

D. Effects of Different Model Architectures and Local Epochs

To evaluate the effects of different model architectures and local epochs on FedALL's performance, we conducted additional experiments using MobileNetV2 and EfficientNet-B0 with one, five, and ten local epochs [38], [39]. We selected these models because they are widely used architectures suitable for federated learning scenarios. Experiments were performed on the CIFAR-10 dataset under the 10–500 participation pattern with $Dir(0.1)$ distribution, representing a challenging scenario with low-client participation and high-data heterogeneity. The effects of variance reduction are presented in Table V and results of model performance are presented in Table VI.

Across all tasks, FedALL-2 consistently outperforms FedAvg in terms of both accuracy and standard deviation, demonstrating its robustness and effectiveness across different architectures and epoch counts. It achieves the best performance in five out of six tasks for five and ten local epochs, ranking second in the remaining task. Compared to FedAvg, FedALL yields average accuracy improvements of 5.13%, 5.88%, and 5.32% for one, five, and ten local epochs, respectively. As expected, the accuracy improvement is larger with five or ten local epochs than with one, since variance caused by partial participation is more severe with more local epochs.

Regarding the HAR dataset, FedALL-2 improves accuracy over FedAvg by 1.04%, 0.93%, and 1.40% for one, five,

TABLE VI

EFFECTS OF DIFFERENT MODEL ARCHITECTURES AND LOCAL EPOCHS ON THE PERFORMANCE OF FEDALL

Methods	ResNet-18		
	E=1	E=5	E=10
FedAvg	59.01 ±4.33	51.20 ±6.45	45.09 ±6.79
MIFA	57.71 ±4.64	58.22 ±3.37	47.15 ±6.40
FedVARP	58.94 ±4.07	58.09 ±5.62	53.08 ±8.86
FedAU	58.91 ±5.12	50.82 ±6.05	44.81 ±6.76
SCAFFOLD	49.78 ±4.92	53.28 ±5.25	42.68 ±6.66
FedALL-1	54.80 ±7.34	58.54 ±1.99	**54.42 ±1.24**
FedALL-2	**67.35 ±2.62**	**58.33 ±3.46**	53.66 ±3.01
FedALL-2-r2	61.75 ±3.54	54.63 ±4.99	48.05 ±4.89
FedALL-2-r5	62.40 ±2.93	56.98 ±3.89	51.31 ±3.76

Methods	MobileNetV2		
	E=1	E=5	E=10
FedAvg	31.02 ±4.46	29.03 ±4.46	25.49 ±4.23
MIFA	34.75 ±2.00	14.74 ±6.63	10.28 ±2.55
FedVARP	34.53 ±2.23	28.93 ±9.74	12.77 ±4.20
FedAU	25.67 ±4.17	28.31 ±4.19	26.52 ±4.48
SCAFFOLD	23.21 ±3.76	29.46 ±4.00	22.76 ±5.10
FedALL-1	**38.99 ±2.32**	24.76 ±3.25	12.12 ±3.79
FedALL-2	36.78 ±3.79	**34.79 ±3.03**	**28.73 ±4.01**
FedALL-2-r2	32.38 ±4.07	28.95 ±8.93	26.32 ±5.06
FedALL-2-r5	32.87 ±4.27	30.51 ±7.38	27.59 ±6.18

Methods	EfficientNet-B0		
	E=1	E=5	E=10
FedAvg	37.48 ±6.30	35.02 ±6.00	32.42 ±6.45
MIFA	38.13 ±3.25	36.58 ±6.55	11.75 ±3.48
FedVARP	**41.24 ±3.39**	35.21 ±9.13	17.81 ±9.55
FedAU	37.23 ±5.34	34.72 ±6.29	32.08 ±6.27
SCAFFOLD	22.00 ±4.54	21.55 ±4.24	21.10 ±7.75
FedALL-1	36.41 ±6.45	34.71 ±7.60	30.47 ±8.13
FedALL-2	38.77 ±3.45	**39.77 ±4.13**	**35.82 ±4.03**
FedALL-2-r2	38.56 ±4.54	36.98 ±6.09	32.73 ±4.24
FedALL-2-r5	40.52 ±6.01	39.27 ±5.54	35.68 ±3.77

and ten local epochs, respectively. The improvement for five local epochs is smaller than that for one local epoch because FedALL-2 slightly underperforms FedAvg in the specific case of the 7–21 participation pattern with five local epochs. If we exclude this task, the accuracy improvement for five local epochs rises to 1.92%, which aligns with previous observations. These results suggest that, in general, FedALL-2 can achieve great improvements when using more local epochs.

In addition, the variance reduction effects are consistently well across different settings. Table V presents the average variance reduction effects of FedALL-2 compared to FedAvg with respect to different models and datasets. In general, we observe that FedALL-2 has a better variance reduction effect on larger or deeper models, such as ResNet-18 and EfficientNet-B0, than on smaller or shallower models, such as LeNet-5 and MobileNetV2.

E. Sensitivity of Hyperparameter λ

As mentioned earlier, λ controls the balance between the current update and the historical update in the aggregation process of FedALL-2. A larger λ places more emphasis on the current update, while a smaller λ gives more weight to the historical update.

To evaluate the impact of hyperparameter λ in FedALL-2 on model performance, we conducted a sensitivity analysis on

TABLE VII

SENSITIVITY OF λ IN FEDALL-2

EMNIST	*Dir*	Acc (default λ)	Acc (best λ)
10-50	0.5	83.63 ±0.30 (1.0)	83.46±0.29 (1.2)
10-50	0.1	82.04 ±0.52 (1.0)	82.20±0.25 (0.5)
10-100	0.5	83.14 ±0.29 (1.0)	83.04±0.41 (1.1)
10-100	0.1	81.60 ±0.50 (1.0)	81.86±0.33 (0.7)
10-200	0.5	82.72 ±0.37 (1.0)	82.76±0.52 (1.1)
10-200	0.1	81.22 ±0.51 (1.0)	81.27±0.57 (0.7)
10-500	0.5	81.72 ±0.37 (1.0)	81.56±0.48 (1.1)
10-500	0.1	80.67 ±0.44 (1.0)	80.74±0.36 (0.9)

CIFAR-10	*Dir*	Acc (default λ)	Acc (best λ)
10-50	0.5	91.64 ±0.28 (1.0)	91.64±0.28 (1.0)
10-50	0.1	80.91 ±2.18 (1.0)	82.68±1.21 (0.8)
10-100	0.5	90.86 ±0.39 (1.0)	90.87±0.44 (0.9)
10-100	0.1	78.09 ±2.83 (1.0)	78.09±2.83 (1.0)
10-200	0.5	88.56 ±1.02 (1.0)	88.59±0.57 (0.9)
10-200	0.1	73.00 ±3.62 (1.0)	74.77±1.88 (0.5)
10-500	0.5	84.83 ±1.24 (1.0)	85.00±0.70 (0.8)
10-500	0.1	67.05 ±2.62 (1.0)	66.91±1.87 (0.9)

CIFAR-100	*Dir*	Acc (default λ)	Acc (best λ)
10-50	0.5	70.66 ±0.40 (1.0)	70.83±0.30 (1.1)
10-50	0.1	66.10 ±1.34 (1.0)	66.34±1.10 (1.2)
10-100	0.5	69.78 ±0.74 (1.0)	69.17±0.47 (0.7)
10-100	0.1	67.37 ±0.62 (1.0)	67.27±1.28 (1.4)
10-200	0.5	68.34 ±0.84 (1.0)	68.67±0.98 (1.2)
10-200	0.1	65.26 ±0.92 (1.0)	65.96±0.96 (1.1)
10-500	0.5	67.78 ±0.75 (1.0)	67.44±0.59 (0.8)
10-500	0.1	63.95 ±1.07 (1.0)	64.58±0.54 (0.9)

HAR	Epoch	Acc (default λ)	Acc (best λ)
7-21	1	90.02 ±0.94 (1.0)	90.02 ±0.94 (1.0)
7-21	5	88.12 ±0.58 (1.0)	88.87 ±1.16 (0.5)
7-21	10	86.60 ±1.16 (1.0)	87.02 ±1.09 (1.2)
3-21	1	89.55 ±0.88 (1.0)	90.26 ±0.79 (0.8)
3-21	5	87.63 ±1.26 (1.0)	87.43 ±0.93 (1.2)
3-21	10	84.77 ±2.01 (1.0)	85.12 ±1.45 (0.9)

CIFAR-10	Epoch	Acc (default λ)	Acc (best λ)
ResNet-18	1	63.95 ±1.07 (1.0)	64.58 ±0.54 (0.9)
ResNet-18	5	58.33 ±3.46 (1.0)	58.17 ±2.93 (0.9)
ResNet-18	10	53.66 ±3.01 (1.0)	55.01 ±2.47 (0.8)
MobileNetV2	1	36.78 ±3.79 (1.0)	37.31 ±3.28 (0.6)
MobileNetV2	5	34.79 ±3.03 (1.0)	35.87 ±3.41 (0.7)
MobileNetV2	10	28.73 ±4.01 (1.0)	28.49 ±4.01 (1.1)
EfficientNet-B0	1	38.77 ±3.45 (1.0)	38.77 ±3.45 (1.0)
EfficientNet-B0	5	39.77 ±4.13 (1.0)	43.01 ±3.01 (0.6)
EfficientNet-B0	10	35.82 ±4.03 (1.0)	37.36 ±3.43 (0.8)

λ over the range $\{0.5, 0.6, 0.7, 0.8, \ldots, 1.4, 1.5\}$, following the same tuning protocol used for learning rates. In Table VII, we present FedALL-2 with default $\lambda = 1$ and with the best λ tuned on benchmark datasets, different local epochs, and model architectures.

We found that the performance gains from tuning λ within the range $\{0.5, 0.6, 0.7, 0.8, \ldots, 1.4, 1.5\}$ are marginal across different datasets, local epochs, and model architectures. In most cases, the accuracy improvement is less than 0.8%, and in some instances, tuning λ even leads to a performance drop. This occurs because the optimal λ values are chosen based on training loss, which may not always correlate perfectly with test accuracy. On the other hand, these results suggest that tuning λ can still provide some performance benefits in certain scenarios and may be worth considering when computational resources permit. In particular, we observe a significant per-

TABLE VIII
STORAGE COSTS OF DIFFERENT METHODS COMPARED TO FEDAVG

Methods	10-50	10-100	10-200	10-500
FedAvg	1.0×	1.0×	1.0×	1.0×
MIFA	5.0×	10.0×	20.0×	50.0×
FedVARP	5.0×	10.0×	20.0×	50.0×
FedAU	1.0×	1.0×	1.0×	1.0×
SCAFFOLD	2.0×	2.0×	2.0×	2.0×
FedAvgAU	1.0×	1.0×	1.0×	1.0×
FedALL-1	5.0×	10.0×	19.9×	49.5×
FedALL-2	5.0×	10.0×	19.9×	49.5×
FedALL-2-r2	1.8×	1.9×	2.0×	2.0×
FedALL-2-r5	3.4×	4.1×	4.5×	4.9×

TABLE IX
WALL-CLOCK TIME (S) OF THE MODEL AGGREGATION PROCESS

Methods	ResNet-18	MobileNetV2	EfficientNet-B0	LSTM
FedAvg	0.065	0.043	0.058	0.002
MIFA	3.273	2.175	2.837	0.116
FedVARP	0.266	0.160	0.216	0.009
SCAFFOLD	0.113	0.025	0.039	0.003
FedALL-1	3.356	2.355	2.942	0.141
FedALL-2	3.267	2.187	2.759	0.106
FedALL-2-r2	0.130	0.079	0.102	0.003
FedALL-2-r5	0.327	0.198	0.256	0.008

formance gain of 3.24% on EfficientNet-B0 with five local epochs. In addition, on EfficientNet-B0 with ten local epochs and the CIFAR-10 dataset with one local epoch, we also observe performance gains of more than 1.5% from tuning λ.

Furthermore, the optimal λ values across different datasets, local epochs, and model architectures are consistently close to 1.0 in most cases. This suggests that $\lambda = 1$ serves as a robust default value and effective starting point for practical applications. This is particularly beneficial in real-world scenarios where extensive hyperparameter tuning may not be feasible due to limited computational resources or time constraints.

F. Memory and Aggregation Efficiency

We evaluate the memory and aggregation efficiency of different methods by comparing their storage and computational costs relative to FedAvg. Storage cost is measured by the actual server storage cost, while aggregation efficiency is measured by the wall-clock time of the model aggregation process on the server, using ten selected clients out of 500. Table VIII presents the average storage costs of different methods relative to FedAvg over 3000 communication rounds. Table IX presents the wall-clock aggregation time for different methods across various model architectures.

From Table VIII, we observe that FedALL-2-r2 and FedALL-2-r5 effectively reduce the storage costs of FedALL-2. Considering their performance on benchmark datasets, they achieve a relatively good tradeoff between performance and storage cost, making them potential alternatives in storage-constrained scenarios. Although they still require more storage than FedAvg, they significantly reduce overhead compared to the original FedALL-2, with only 1.8× and 3.4× storage costs compared to FedAvg, which is more manageable in practice. On the contrary, MIFA and FedVARP require the server to store historical updates for all clients: MIFA uses these directly for model aggregation, while FedVARP uses them to update two auxiliary vectors for model aggregation.

Regarding aggregation efficiency, the aggregation time of FedALL-2 and MIFA scales with the total number of clients, as expected, since they both store and aggregate historical updates for all clients. We also observe that FedALL-1 slightly increases aggregation time compared to FedALL-2. While it requires extra time to compute the trace of variance and covariance matrices, this overhead accounts for 2.5% of the total aggregation time of FedALL-1.

G. Summary

In summary, both FedALL-1 and FedALL-2 outperform other methods overall, with FedALL-2 achieving the best performance. In particular, FedALL-2 improves average accuracy by 2.51% and reduces average standard deviation by 0.74% compared to FedAvg across four benchmark datasets. This leads to a 1.85% accuracy improvement over the second-best method, FedVARP. Notably, FedALL-2 wins 21 out of 30 tasks, while FedALL-1, SCAFFOLD, and MIFA win four, three, and two tasks, respectively. Considering the storage-efficient variant, FedALL-2-r5 even outperforms FedALL-2 in 1 out of the 21 tasks won by the latter. In addition, FedALL-2 outperforms the baseline FedAvg in 29 out of 30 tasks, highlighting its robustness across different scenarios. The only exception is the 7–21 participation pattern with five local epochs on the HAR dataset, where FedALL-2 shows a slight performance drop compared to FedAvg.

In scenarios with limited storage, FedALL-2-r2 and FedALL-2-r5 serve as effective alternatives that significantly reduce storage costs. Among all methods, FedALL-2-r5 outperforms other approaches except FedALL-2, while FedALL-2-r2 remains competitive. These results indicate that both variants are effective options when server storage is constrained. Furthermore, FedALL-2 exhibits minimal sensitivity to the hyperparameter λ. $\lambda = 1$ performs consistently well across different datasets, local epochs, and model architectures, making it a robust default choice. In specific cases, tuning λ yields additional benefits with performance gains up to 3.24%.

VI. CONCLUDING REMARKS

In this article, we systematically studied the partial participation problem in federated learning. We first explained the rationale for leveraging clients' most recent updates—a common approach in previous research—by examining the bias and variance during the model aggregation. Our analysis demonstrates that effectively using clients' most recent updates can reduce the variance of the aggregation step. To fully exploit this advantage, we propose a new algorithm, FedALL, which aims to minimize variance while maintaining a small bias. Extensive experiments confirm the superior performance of FedALL, with both FedALL-1 and FedALL-2 outperforming other methods. FedALL-2 excels in both accuracy and standard deviation, improving average accuracy by 2.51% and decreasing average standard deviation by 0.74%. This results in a 1.85% improvement in accuracy and a 0.6% improvement in standard deviation compared to the previous state-of-the-art method, FedVARP. Notably, FedALL-2 achieves this with default hyperparameter $\lambda = 1$. Empirical results show that $\lambda = 1$ is near-optimal across different datasets, local epochs, and model architectures, with tuning normally yielding

gains of less than 0.5% in accuracy. This robustness makes FedALL-2 easy to deploy without extensive hyperparameter tuning. In addition, for servers with limited storage, we introduced an optional storage-efficient module for FedALL, which reduces server storage costs by caching updates from only the most recent p rounds. Our experiments show that these variants remain competitive with FedVARP.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[2] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[3] P. Kairouz and H. B. McMahan, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, Jun. 2021.

[4] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, Mar. 2021, Art. no. 106775.

[5] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. Vincent Poor, "Federated learning for Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1622–1658, 3rd Quart., 2021.

[6] X. Gu, K. Huang, J. Zhang, and L. Huang, "Fast federated learning in the presence of arbitrary device unavailability," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 12052–12064. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/64be20f6dd1dd46adf110cf871e3ed35-Paper.pdf

[7] D. Jhunjhunwala, P. Sharma, A. Nagarkatti, and G. Joshi, "FedVARP: Tackling the variance due to partial client participation in federated learning," in *Proc. 38th Conf. Uncertain. Artif. Intell.*, 2022, pp. 906–916. [Online]. Available: https://proceedings.mlr.press/v180/jhunjhunwala22a.html

[8] V. Saligrama, D. A. E. Acar, P. N. Whatmough, R. Matas, M. Mattina, and Y. Zhao, "Federated learning based on dynamic regularization," in *Proc. 9th Int. Conf. Learn. Represent. (ICLR)*, Austria, May 2021. [Online]. Available: https://openreview.net/forum?id=B7v4QMR6Z9w

[9] Y. Jee Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *Proc. 25th Int. Conf. Artif. Intell. Statist.*, vol. 151, pp. 10351–10375. [Online]. Available: https://proceedings.mlr.press/v151/jee-cho22a.html

[10] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. ICC - IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.

[11] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang, and H. Vincent Poor, "Fast-convergent federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 201–218, Jan. 2021.

[12] G. Liao, B. Luo, Y. Feng, M. Zhang, and X. Chen, "Optimal mechanism design for heterogeneous client sampling in federated learning," *IEEE Trans. Mobile Comput.*, vol. 23, no. 11, pp. 10598–10609, Nov. 2024.

[13] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu, "Client selection in federated learning: Principles, challenges, and opportunities," *IEEE Internet Things J.*, vol. 10, no. 24, pp. 21811–21819, Dec. 2023.

[14] Y. Yan et al., "Federated optimization under intermittent client availability," *INFORMS J. Comput.*, vol. 36, no. 1, pp. 185–202, Jan. 2024, doi: 10.1287/ijoc.2022.0057.

[15] M. Ji and S. Wang, "A unified analysis of federated learning with arbitrary client participation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 19124–19137. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/79ba1b827d3fc58e129d1cbfc8ff69f2-Paper-Conference.pdf

[16] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. 3rd Mach. Learn. Syst. Conf.*, 2020, pp. 429–450.

[17] S. Wang and M. Ji, "A lightweight method for tackling unknown participation statistics in federated averaging," in *Proc. 12th Int. Conf. Learn. Represent. (ICLR)*. Vienna, Austria: OpenReview.net, May 2023. [Online]. Available: https://dblp.org/rec/conf/iclr/0001J24.bib

[18] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, p. 1646.

[19] M. Schmidt, N. Le Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," *Math. Program.*, vol. 162, nos. 1–2, pp. 83–112, Mar. 2017.

[20] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 315–323. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2013/file/ac1dd209cbcc5e5d1c6e28598e8cbbe8-Paper.pdf

[21] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. 37th Int. Conf. Mach. Learn.*, vol. 119, Jul. 2020, pp. 5132–5143. [Online]. Available: https://proceedings.mlr.press/v119/karimireddy20a.html

[22] B. Li, M. N. Schmidt, T. S. Alstrøm, and S. U. Stich, "On the effectiveness of partial variance reduction in federated learning with heterogeneous data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 3964–3973.

[23] Y. Jee Cho, S. Gupta, G. Joshi, and O. Yagan, "Bandit-based communication-efficient client selection strategies for federated learning," in *Proc. 54th Asilomar Conf. Signals, Syst., Comput.*, Nov. 2020, pp. 1066–1069.

[24] H. Yang, M. Fang, and J. Liu, "Achieving linear speedup with partial worker participation in non-iid federated learning," in *Proc. 9th Int. Conf. Learn. Represent. (ICLR)*, Austria, May 2021. [Online]. Available: https://openreview.net/forum?id=jDdzh5ul-d

[25] F. Wu, S. Guo, Z. Qu, S. He, Z. Liu, and G. Jing, "Anchor sampling for federated learning with partial client participation," in *Proc. 40th Int. Conf. Mach. Learn.*, 2022, pp. 37379–37416.

[26] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-IID data silos: An experimental study," in *Proc. IEEE 38th Int. Conf. Data Eng. (ICDE)*, May 2022, pp. 965–978.

[27] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "A novel framework for the analysis and design of heterogeneous federated learning," *IEEE Trans. Signal Process.*, vol. 69, pp. 5234–5249, 2021.

[28] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," 2019, *arXiv:1907.02189*.

[29] S. J. Reddi et al., "Adaptive federated optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*. Austria: OpenReview.net, May 2021. [Online]. Available: https://openreview.net/forum?id=LkFG3lB13U5

[30] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," M. Sc thesis, Dept. Comput. Sci., University of Toronto, Toronto, ON, Canada, 2009.

[31] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: Extending MNIST to handwritten letters," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 2921–2926.

[32] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. Eur. Symp. Artif. Neural Netw.*, 2013, pp. 437–442. [Online]. Available: https://api.semanticscholar.org/CorpusID:6975432

[33] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017, *arXiv:1412.6980*.

[37] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proc. 36th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds., PMLR, Jun. 2019. [Online]. Available: http://proceedings.mlr.press/v97/yurochkin19a/yurochkin19a.pdf

[38] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," 2019, *arXiv:1801.04381*.

[39] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, Jun. 2019, pp. 6105–6114.