FedCAD: Federated Learning with Clustering, Adaptive Selection, and Delayed Aggregation for Heterogeneous IoT Environments

Tian Liu, *Member, IEEE,* Zhiwei Ling, Ziqi Wang, Jiahui Zhai, Chenggang Shan, Zhen Yang, and Bin Yang, *Member, IEEE*

***Abstract*—Federated Learning (FL) is a critical enabler for intelligent Internet of Things (IoT) systems, allowing collaborative model training across heterogeneous devices while preserving data privacy. However, FL performance degrades significantly under non-Independent and Identically Distributed (non-IID) data due to weight divergence, and existing mitigation methods often introduce substantial overhead unsuitable for resource-constrained IoT deployments. We identify that weight divergence in non-IID FL stems primarily from insufficient cross-device knowledge exchange before aggregation—a contributing factor that has been underexplored in existing literature—and propose FedCAD, a communication-efficient FL framework that directly addresses this root cause. The core innovation is delayed cross-group aggregation, which strategically postpones global model updates until each model copy has been sequentially trained across all device groups, ensuring comprehensive knowledge integration. This mechanism is supported by dual-feature device clustering that creates meaningful group structures and fairness-aware adaptive selection that ensures representative participation—forming a causally linked pipeline where clustering provides structure, selection provides quality, and delayed aggregation provides the mechanism for thorough knowledge exchange. Extensive experiments on five benchmark datasets across 13 heterogeneous scenarios demonstrate that FedCAD consistently outperforms ten state-of-the-art methods with negligible additional communication overhead.**

***Index Terms*—Federated Learning, Internet of Things, Data Heterogeneity, Delayed Aggregation, Device Clustering.**

I. Introduction

THE rapid expansion of the Internet of Things (IoT) has created a vast ecosystem of heterogeneous edge devices—including smart sensors, wearable monitors, industrial controllers, and autonomous vehicles—that continuously generate massive volumes of data at the network edge [1], [2]. Training intelligent models on this distributed data is essential for enabling real-time decision-making in IoT applications such as predictive maintenance, health monitoring, and smart transportation. Federated Learning (FL) has emerged as the predominant paradigm for collaborative model training in IoT environments, allowing distributed devices to jointly learn a shared global model without centralizing raw data, thereby preserving data privacy and reducing communication bandwidth [3]. However, deploying FL in real-world IoT systems faces a fundamental challenge: the data collected by IoT devices is inherently non-Independent and Identically Distributed (non-IID) due to differences in device types, deployment locations, user behaviors, and operating conditions [4], [5].

This data heterogeneity poses a severe challenge to FL performance. In non-IID scenarios, where each device possesses data from different underlying distributions, model performance degrades substantially due to the well-known "weight divergence" problem. This occurs because local models trained on heterogeneous data converge toward different optima [6], causing the aggregated global model to perform poorly on individual devices. Furthermore, theoretical analysis has shown that data heterogeneity can severely impede convergence speed [6], making it difficult to achieve satisfactory model quality within a reasonable number of training rounds.

To tackle the performance degradation in non-IID settings, researchers have proposed various FL approaches, including global control variable techniques [7], [8], and Knowledge Distillation (KD) methods [9]–[11]. While these approaches demonstrate improvements in classification accuracy, they frequently come with drawbacks such as increased communication costs, additional computational burden on resource-limited devices, or potential privacy leakage risks—all of which are particularly problematic for resource-constrained IoT deployments. More fundamentally, we observe that these methods address the *symptoms* of weight divergence (e.g., by regularizing local updates or distilling knowledge) rather than its *root cause*: in standard FL, the global model is aggregated every round from a small subset of devices, providing insufficient exposure to the full diversity of data distributions before each aggregation step. This premature aggregation forces the global model to reconcile conflicting gradient directions without adequate cross-device knowledge exchange, leading to suboptimal convergence. *Consequently, developing an FL framework that addresses the root cause of weight divergence—insufficient knowledge exchange before aggregation—while avoiding additional device-side overhead remains a critical challenge for practical IoT systems.*

In response to this challenge, we propose FedCAD, a novel

This work was supported by the Natural Science Foundation of Shandong Province (NO. ZR2023QF131, ZR2023MF061, ZR2025MS1036, and ZR2025LH117). (Corresponding authors: Bin Yang and Chenggang Shan.)

Tian Liu, Chenggang Shan, Zhen Yang, and Bin Yang are with the School of Information Science and Engineering, Zaozhuang University, Zaozhuang, 277160, China (e-mail: liutian@uzz.edu.cn; shanchenggang@uzz.edu.cn; yang_zh99@163.com; batsi@126.com).

Zhiwei Ling and Ziqi Wang are with the School of Software Technology, Zhejiang University, China (e-mail: zwling@zju.edu.cn; wangziqi0312@zju.edu.cn).

Jiahui Zhai is with the College of Computer Science, Beijing University of Technology, China (e-mail: zhaijiahui@emails.bjut.edu.cn).

TABLE I
COMPARISON OF FEDCAD WITH REPRESENTATIVE FL METHODS FOR NON-IID DATA.

Method	Category	Key Strength	Main Limitation	Extra Overhead
FedAvg [3]	Baseline	Simple, efficient	Poor non-IID performance	None
FedProx [12]	Adaptive Opt.	Stabilizes training	Limited heterogeneity	Device-side computation
SCAFFOLD [7]	Adaptive Opt.	Reduces client drift	Additional memory	Device-side computation
FedGen [10]	Knowledge Dist.	Improves accuracy	High communication cost	Generator transmission
FedDF [9]	Knowledge Dist.	Ensemble learning	Privacy concerns	Auxiliary dataset
CluSamp [15]	Device Grouping	Reduces variance	Extra communication	Clustering overhead
FedCCFA [16]	Device Grouping	Classifier clustering + feature alignment for concept drift	Requires anchor generation per round	Feature anchor computation
MOON [17]	Contrastive Learning	Aligns updates	Computational overhead	Contrastive loss
FedCAD (Ours)	**Delayed Cross-Group Aggregation**	**Root-cause mitigation of weight divergence**	**Periodic re-clustering**	**Clustering (negligible, server-side)**

federated learning framework specifically designed for heterogeneous IoT environments. The core innovation of FedCAD is a **delayed cross-group aggregation** mechanism that directly addresses the root cause of weight divergence identified above. Instead of aggregating models every round, FedCAD strategically postpones global model updates, allowing each model copy to be sequentially trained by devices from *every* group before aggregation. This ensures comprehensive cross-group knowledge exchange, enabling the global model to learn from the full spectrum of data distributions before each update.

To maximize the effectiveness of delayed aggregation, FedCAD incorporates two supporting mechanisms that form a coherent pipeline: (1) **dual-feature device clustering** groups IoT devices with similar data characteristics and model behaviors, creating meaningful groups for the cross-group rotation; and (2) **fairness-aware adaptive selection** ensures that representative devices are chosen from each group in every round, preventing participation bias. These three components are not independent techniques but form a *causally linked chain*: clustering creates the group structure that delayed aggregation requires, and adaptive selection ensures each rotation step uses high-quality representatives. The main contributions of this work are:

- We identify that insufficient cross-device knowledge exchange before aggregation is a primary contributing factor to weight divergence in non-IID FL, and propose a novel **delayed cross-group aggregation** mechanism that directly addresses this issue by postponing aggregation until each model copy has been trained across all device groups.
- We develop a lightweight **dual-feature device clustering** mechanism and a **fairness-aware adaptive selection** strategy as causally linked supporting components—clustering creates meaningful group structures for cross-group rotation, while adaptive selection ensures representative participation within each group.
- We provide rigorous theoretical convergence analysis demonstrating that FedCAD achieves a tighter convergence bound than FedAvg, and validate our framework through extensive experiments on five benchmark datasets across 13 heterogeneous scenarios.

Our experimental evaluation shows that FedCAD achieves superior accuracy and convergence compared to ten state-of-the-art FL methods (including FedAvg), while introducing only negligible additional communication overhead.

II. RELATED WORK

While FL offers benefits in communication efficiency and privacy preservation compared to traditional distributed learning, achieving high model accuracy remains challenging when training data is heterogeneously distributed across devices. Comprehensive surveys of FL for IoT systems [18], [19] highlight that data heterogeneity, resource constraints, and device reliability are among the most critical open challenges in this domain. Existing solutions to this problem can be broadly categorized into three approaches: knowledge distillation techniques, adaptive optimization with global control variables, and device grouping strategies.

Knowledge distillation approaches transfer knowledge from teacher models to student models using soft labels. FedGen [10] employs a generator-based distillation strategy that operates without requiring actual data samples, addressing heterogeneity through synthetic data generation. FedDF [9] aggregates knowledge from local models by performing ensemble distillation on unlabeled datasets at the server. Despite their effectiveness, these techniques often necessitate either transmitting generator models or maintaining auxiliary datasets, leading to increased communication costs and raising privacy concerns.

Adaptive optimization methods employ global control variables or modified update rules to enhance convergence stability. FedProx [12] adds a proximal regularization term to local objectives, constraining model drift from the global model. SCAFFOLD [7] uses control variates to compensate for client drift across heterogeneous devices. Adaptive optimizers such as FedAdam and FedYogi [14] incorporate momentum-based learning rate adaptation to improve convergence. FedNova [13] normalizes local updates to handle objective inconsistency caused by varying local data sizes. While these methods improve stability, they can incur additional computational costs and show limited gains in severely heterogeneous environments.

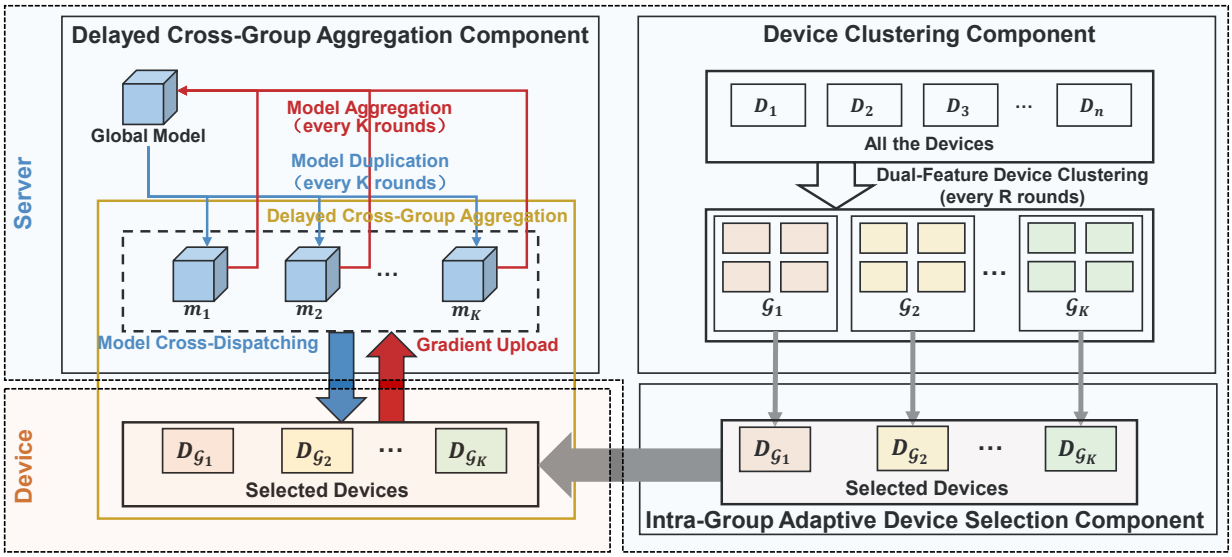


Fig. 1. Architectural overview of the FedCAD framework.

Device grouping, personalization, and feature alignment strategies organize devices based on data or model similarity to mitigate heterogeneity effects. CluSamp [15] clusters devices by data volume or model similarity to reduce aggregation variance and improve representativeness. MOON [17] applies model-contrastive learning to align local model updates and reduce drift. More recently, FedCCFA [16] (addresses data heterogeneity under distributed concept drift by clustering local classifiers at the class level and generating clustered feature anchors to adaptively align clients' feature spaces, demonstrating strong performance when client data distributions shift over time. These approaches, however, typically require extra communication rounds for clustering or additional per-round overhead for anchor generation, potentially hindering scalability in resource-constrained IoT environments.

Federated Learning for IoT. The deployment of FL in IoT environments introduces unique challenges beyond data heterogeneity, including resource constraints, intermittent connectivity, and large-scale device management [1], [2], [11]. Recent works have explored various aspects of FL for IoT: Meng et al. [1] proposed a generative AI-based FL algorithm to address extreme data and resource heterogeneity across IoT devices. Zhou et al. [2] designed a knowledge fusion scheme to mitigate client model drift caused by heterogeneous IoT data. Liu et al. [11] developed knowledge distillation-based FL for AIoT applications, reducing model transmission costs. However, these approaches primarily focus on data augmentation, knowledge transfer, or model compression, without addressing the fundamental challenge of how to effectively aggregate knowledge from heterogeneous IoT devices with diverse data distributions. FedCAD fills this gap by providing a principled aggregation framework that enables comprehensive cross-device knowledge exchange before model updates, making it particularly suitable for large-scale IoT deployments where device heterogeneity is the norm rather than the exception.

FedCAD distinguishes itself from existing approaches through a causally motivated design: rather than treating clustering, selection, and aggregation as independent techniques, FedCAD derives its architecture from a root-cause analysis of weight divergence, resulting in a coherent framework where each component serves a necessary role. Table I provides a systematic comparison of FedCAD with representative methods from each category. This principled design enables FedCAD to deliver superior performance while introducing only minimal additional communication overhead for periodic feature extraction.

III. The Proposed FedCAD Framework

The global and local optimization objectives of FedCAD are defined as follows:

$$\begin{aligned} &\min_{w} F(w) = \sum_{i=1}^{N} p_i f_i(w), \\ &s.t.,\ p_i = \frac{n_i}{\sum_{i=1}^{N} n_i}, f_i(w) = \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(w; x_j; y_j), \end{aligned} \tag{1}$$

where N is the total number of devices, p_i is the weight of the i^{th} device that depends on the number of data samples (i.e., n_i) in the i^{th} device, ℓ denotes a kind of user-defined loss function (e.g. cross-entropy loss).

A. FedCAD Framework and Workflow

Fig. 1 shows the architecture of FedCAD, which consists of three components: Device Clustering, Intra-Group Adaptive Selection, and Delayed Cross-Group Aggregation. Unlike conventional FL methods that aggregate models every round, FedCAD operates on a hierarchical two-level cycle structure: a major regrouping cycle of R rounds and a minor delayed aggregation cycle of K rounds, where K is the number of device groups.

1) Overall Workflow: The overall workflow of FedCAD is formalized in Algorithm 1. The process begins with server-side random initialization of the global model (line 1). The main training loop (lines 3-9) proceeds for T rounds. At the beginning of each major regrouping cycle (when $t \bmod R = 0$), the server updates device features and re-partitions the devices into K groups using Algorithm 2 (lines 4-6). Training then occurs in minor cycles of K rounds. At the start of each minor cycle (when $t \bmod K = 0$), a full delayed cross-group aggregation is executed as detailed in Algorithm 4, after which the main round counter t is advanced by K steps to reflect the completion of the cycle (lines 7-9).

Algorithm 1 The FedCAD Algorithm

Input: Total devices N, total rounds T, regrouping period R, number of groups K, local epochs E.
Output: Final global model w_T.

1: Initialize global model w_0 randomly and device participation counts $\text{count}_i = 0$ for $i = 1, ..., N$.
2: Server extracts and caches initial data distribution features $\{\mathbf{d}_i\}_{i=1}^{N}$.
3: **for** $t = 0, 1, ..., T-1$ **do**
4: **if** $t \bmod R = 0$ **then**
5: Server extracts model similarity features $\{\mathbf{m}_i^t\}_{i=1}^{N}$.
6: Partition devices into K groups $\{\mathcal{G}_k\}_{k=1}^{K}$ using Algorithm 2($\mathcal{C}$, $\{\mathbf{d}_i\}$, $\{\mathbf{m}_i^t\}$, K).
7: **end if**
8: **if** $t \bmod K = 0$ **then**
9: Execute one full cycle of delayed cross-group aggregation (Algorithm 4($\{\mathcal{G}_k\}_{k=1}^{K}$, w_t, t, E, counts)) to get w_{t+K}.
10: $t \leftarrow t + K - 1$ {Advance t to the end of the cycle}
11: **end if**
12: **end for**
13: **return** w_T.

2) Device Clustering Component: To capture both static data properties and dynamic training behavior, devices are grouped based on two distinct features. This process, performed every R rounds, is detailed in Algorithm 2.

Feature Extraction: The server coordinates the extraction of two complementary feature vectors for each device, designed to capture both static data characteristics and dynamic training behavior:

- **Data Distribution Features** ($\mathbf{d}_i$): A C-dimensional vector characterizing the statistical properties of a device's local dataset. In supervised settings, this can be the class frequency distribution; in unsupervised or label-scarce IoT scenarios, this can be approximated using the model's output prediction distribution (i.e., the average softmax output over local data), which does not require ground-truth labels. This feature is extracted once at $t = 0$ and cached, with negligible privacy impact as only aggregate statistics are transmitted.
- **Model Similarity Features** ($\mathbf{m}_i$): A fixed-dimensional vector (obtained through PCA dimensionality reduction) summarizing statistical properties (mean, standard deviation, maximum, minimum, median) of model parameters across different layers. This feature is updated every R rounds to capture the evolving training dynamics. Notably, this feature is *label-independent* and serves as the *primary* clustering signal, as it directly reflects the model's learned representation of local data characteristics.

Algorithm 2 Dual-Feature Device Clustering in FedCAD

Input: Device set $\mathcal{C}$, cached data features $\{\mathbf{d}_i\}$, current model features $\{\mathbf{m}_i\}$, number of clusters K.
Output: A partition of $\mathcal{C}$ into K groups $\{\mathcal{G}_k\}_{k=1}^{K}$.

1: Initialize feature list $\mathcal{F} = \emptyset$.
2: **for** each device $i \in \mathcal{C}$ **do**
3: **if** model features $\{\mathbf{m}_i\}$ are available **then**
4: $\mathbf{f}_i \leftarrow \text{concat}(\mathbf{d}_i, \mathbf{m}_i)$.
5: **else**
6: $\mathbf{f}_i \leftarrow \mathbf{d}_i$.
7: **end if**
8: Add $\mathbf{f}_i$ to $\mathcal{F}$.
9: **end for**
10: Preprocess $\mathcal{F}$ using StandardScaler and PCA.
11: Apply KMeans to $\mathcal{F}$ to get clusters $\mathcal{P}_K$ and compute $\text{score}_K \leftarrow \text{silhouette}(\mathcal{P}_K)$.
12: **if** $\text{score}_K < 0.3$ **then**
13: Apply DBSCAN to get clusters $\mathcal{P}_D$, compute score_D, and use $\mathcal{P}_D$ if $\text{score}_D > \text{score}_K$.
14: **end if**
15: Balance clusters by merging small groups.
16: **return** balanced clusters as $\{\mathcal{G}_k\}_{k=1}^{K}$.

As shown in Algorithm 2, the server first prepares a combined feature vector for each device by concatenating the static data features and the dynamic model features (lines 1-9). Formally, the combined feature vector is defined as:

$$\mathbf{f}_i = \begin{cases} [\mathbf{d}_i; \mathbf{m}_i] & \text{if model features available} \\ \mathbf{d}_i & \text{otherwise} \end{cases} \tag{2}$$

where $[\cdot;\cdot]$ denotes vector concatenation. These combined features are then preprocessed using StandardScaler [20] and dimensionality is reduced via Principal Component Analysis (PCA) [21] to improve clustering quality (line 10).

The algorithm evaluates both KMeans [22] and DBSCAN [23] by computing silhouette scores [20] (lines 11–14). The silhouette score $s(i) = (b(i) - a(i))/\max\{a(i), b(i)\}$ measures clustering quality, where $a(i)$ is the mean intra-cluster distance and $b(i)$ is the mean nearest-cluster distance. If KMeans achieves a silhouette score above 0.3, it is used; otherwise, DBSCAN is applied. Finally, the algorithm balances cluster sizes by merging small groups (line 15).

In this design, the *model similarity features* serve as the primary clustering signal because they are entirely label-independent and naturally capture data heterogeneity through trained model parameters. The *data distribution features* provide a complementary signal; in scenarios where ground-

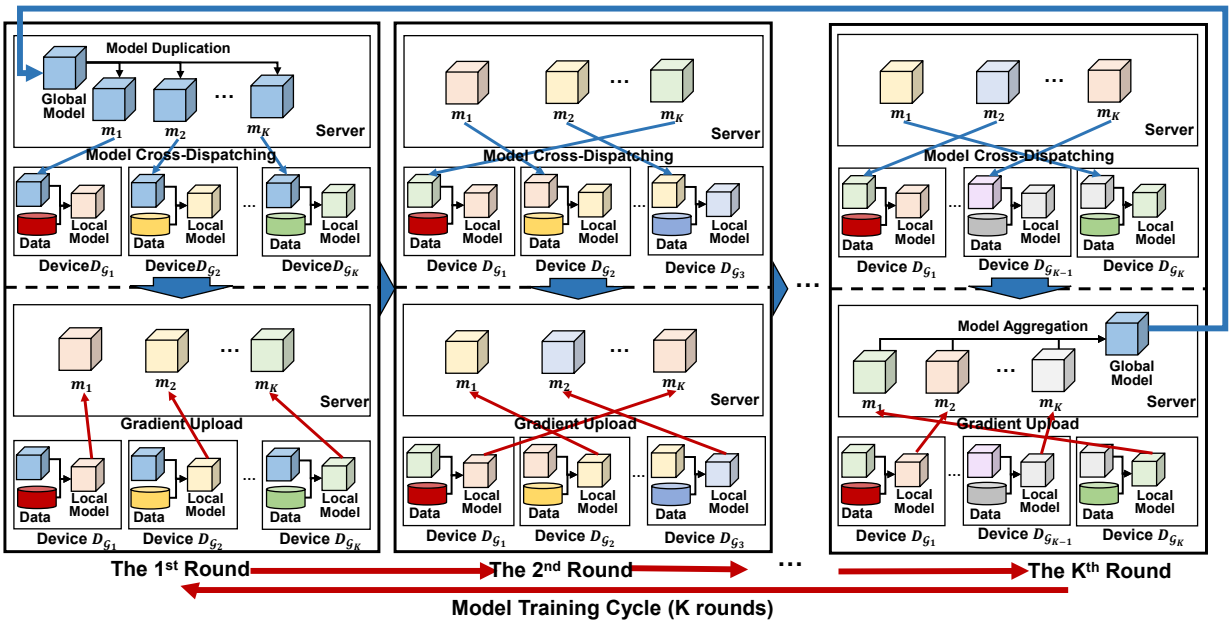


Fig. 2. Workflow of the Delayed Cross-Group Aggregation mechanism in FedCAD.

truth labels are unavailable—common in IoT applications such as anomaly detection—they can be replaced by the model's prediction distribution (average softmax outputs). As confirmed in our ablation study (Section IV-C, FedCAD-C variant), model similarity features alone provide sufficient signal for meaningful device grouping.

3) Intra-Group Adaptive Device Selection Component: Algorithm 3 describes the fair and efficient intra-group selection mechanism used to choose one device from each group per round. The selection probability $P(i)$ for a device i in group $\mathcal{G}_k$ balances fairness and data contribution:

$$P(i) \propto \frac{1}{\sqrt{\text{count}_i + 1}} \times \left(1 + \beta \cdot \frac{n_i}{\max_{j \in \mathcal{G}_k} n_j}\right) \tag{3}$$

where count_i is the historical participation count, n_i is the local data size, and β is a hyperparameter (typically 0.5). The first term promotes fairness, while the second term gives a moderate boost to devices with larger datasets. To ensure valid probability distribution, these values are normalized within each group:

$$\tilde{P}(i) = \frac{P(i)}{\sum_{j \in \mathcal{G}_k} P(j)}, \quad \forall i \in \mathcal{G}_k \tag{4}$$

where $\tilde{P}(i)$ represents the normalized selection probability used for sampling.

The selection process operates independently within each group. For each group, the algorithm computes selection probabilities using Eq. (3), normalizes them via Eq. (4), and samples one device accordingly (lines 2–7). By combining inverse-square-root fairness weighting with data-size awareness, this strategy ensures balanced participation while leveraging devices with richer data.

Algorithm 3 Intra-Group Adaptive Device Selection in FedCAD

Input: Device groups $\{\mathcal{G}_k\}_{k=1}^{K}$, counts $\{\text{count}_i\}$, data sizes $\{n_i\}$, β.
Output: A list of K selected devices $\{c_k\}_{k=1}^{K}$.

1: Initialize selected device list $\mathcal{S} = \emptyset$.
2: **for** each group $\mathcal{G}_k$ from $k = 1, ..., K$ **do**
3: Calculate probabilities $P(i)$ for all $i \in \mathcal{G}_k$ using Eq. (3).
4: Normalize probabilities within the group.
5: Select one device c_k from $\mathcal{G}_k$ based on probabilities.
6: Add c_k to $\mathcal{S}$ and update count_{c_k}.
7: **end for**
8: **return** $\mathcal{S}$.

4) Delayed Cross-Group Aggregation Component: This mechanism facilitates comprehensive knowledge sharing across different device groups. The process is detailed in Algorithm 4 and illustrated in Fig. 2. The rationale behind this design is to directly address the root cause of weight divergence: premature aggregation of conflicting gradients. By separating the system framework (Fig. 1) from the operational workflow (Fig. 2), we clearly illustrate how FedCAD transforms the standard star-topology aggregation into a sequential cross-group pipeline. At the start of a delayed aggregation cycle, the server creates K identical copies of the current global model (line 1). Over the next K rounds (lines 3-11), these model copies are cross-dispatched to selected devices using a deterministic rotational mechanism, formalized as:

$$\text{target\_group}(i, j) = ((i - 1) + (j - t)) \bmod K + 1 \tag{5}$$

where i is the model copy index, j is the current round, and

t is the cycle start round. This ensures each copy is trained by one device from every group. After a device performs local training (line 8), the updated model is sent back to the server (line 9). Finally, after K rounds, an adaptive aggregation is performed (line 13) using data-weighted averaging:

$$w_{t+K} = \sum_{i=1}^{K} \alpha_i \cdot w_{t+K,i}, \quad \text{where} \quad \alpha_i = \frac{D_i}{\sum_{j=1}^{K} D_j} \tag{6}$$

where D_i is the cumulative data size of all devices that trained model copy i, and α_i is the aggregation weight. This ensures that model copies trained on larger datasets have proportionally greater influence on the final aggregated model.

Robustness Mechanism: To handle device failures during training, FedCAD implements a timeout-based fallback strategy where the server automatically selects an alternative device from the same group. If a device drops out mid-training, the aggregation proceeds with the remaining $K-1$ model copies with renormalized weights.

Algorithm 4 Delayed Cross-Group Aggregation in FedCAD

Input: Groups $\{\mathcal{G}_k\}_{k=1}^{K}$, model w_t, start round t, local epochs E.

Output: Updated global model w_{t+K}.

1: Duplicate global model into K copies: $w_{t,i} \leftarrow w_t$ for $i = 1, ..., K$.
2: Initialize cumulative data sizes $D_i \leftarrow 0$ for each copy.
3: **for** round $j = t$ to $t + K - 1$ **do**
4: Select K devices $\{c_k\}_{k=1}^{K}$ using Algorithm 3($\{\mathcal{G}_k\}_{k=1}^{K}$, $\{\text{count}_i\}$, $\{n_i\}$, β).
5: **for** model copy index $i = 1$ to K **do**
6: Compute target group: $k \leftarrow ((i-1) + (j-t)) \bmod K + 1$.
7: Dispatch model copy $w_{j,i}$ to device c_k.
8: Device c_k performs local training: $w_{j,i}^{\text{new}} \leftarrow \text{LocalTrain}(w_{j,i}, \mathcal{D}_{c_k}, E)$.
9: Server receives $w_{j,i}^{\text{new}}$ and updates $w_{j+1,i} \leftarrow w_{j,i}^{\text{new}}$ and $D_i \leftarrow D_i + n_{c_k}$.
10: **end for**
11: **end for**
12: Compute aggregation weights: $\alpha_i \leftarrow D_i / \sum_{j=1}^{K} D_j$.
13: Perform adaptive aggregation: $w_{t+K} \leftarrow \sum_{i=1}^{K} \alpha_i w_{t+K,i}$.
14: **return** w_{t+K}.

In summary, the delayed aggregation cycle creates K model copies (line 1), dispatches them to devices from different groups via rotational scheduling (lines 3–11), and performs data-weighted aggregation after all copies have been trained across all groups (lines 12–13). This design ensures comprehensive cross-group knowledge integration before each global model update.

Relationship to Ring-Topology Approaches: FedCAD's rotational dispatch extends the ring-topology paradigm [29] from a communication-level primitive to a training-level strategy. While Ring-AllReduce optimizes *how* gradients are communicated without altering the training process, FedCAD fundamentally changes *what* is aggregated and *when*: it maintains K independent model copies that each traverse all device groups before aggregation, transforming the aggregation step from simple parameter averaging into structured cross-group knowledge fusion.

B. Theoretical Analysis

1) Communication Complexity Analysis: We provide a rigorous analysis of FedCAD's communication complexity and compare it with baseline methods.

Theorem 1 (Communication Complexity of FedCAD)**.** *Let d denote the model parameter dimension, K the number of device groups, N the total number of devices, T the total training rounds, and R the regrouping period. The total communication cost of FedCAD over T rounds is:*

$$C_{\textit{FedCAD}} = 2Kd \cdot T + f \cdot N \cdot \lceil T/R \rceil, \tag{7}$$

where f is the feature vector dimension and $\lceil \cdot \rceil$ denotes the ceiling function. The first term represents model parameter transmission, and the second term represents periodic feature extraction overhead.

Proof. We decompose the communication cost into two parts: model parameter transmission and feature extraction for regrouping.

Part 1: Model parameter transmission. During each training round, FedCAD selects K devices (one per group) through the adaptive selection mechanism described in Algorithm 3. Each selected device receives a model copy from the server, requiring Kd parameters to be downloaded (where d is the model dimension). After performing E local training epochs, each device sends back updated parameters, requiring Kd parameters to be uploaded. Thus, the bidirectional communication per round is $2Kd$ parameters. Over T training rounds, this yields $2Kd \cdot T$ total communication for model parameters.

Part 2: Feature extraction for regrouping. Every R rounds, the server triggers a regrouping operation (Algorithm 2) to adapt to evolving device characteristics. During regrouping, all N devices transmit feature vectors of dimension f to the server. Each device sends one feature vector of size f, resulting in Nf parameters transmitted per regrouping. Over T rounds, regrouping occurs $\lceil T/R \rceil$ times, contributing $f \cdot N \cdot \lceil T/R \rceil$ to the total cost.

Total communication cost. Combining both parts yields the stated complexity: $C_{\text{FedCAD}} = 2Kd \cdot T + f \cdot N \cdot \lceil T/R \rceil$. □

Comparison with FedAvg: FedAvg's communication cost is $C_{\text{FedAvg}} = 2md \cdot T$, where m is the number of selected devices per round. When $K = m$ (same participation rate), the per-round cost is identical. The additional feature extraction cost in FedCAD is:

$$\Delta C = f \cdot N \cdot \lceil T/R \rceil \approx \frac{f \cdot N \cdot T}{R}. \tag{8}$$

Since $f \ll d$ (feature dimension is much smaller than model dimension) and R is typically large (e.g., 100), the amortized overhead per round is $\frac{f \cdot N}{R} \ll Kd$, making it negligible in practice.

2) Convergence Analysis: The convergence of FedCAD is established under standard assumptions for non-convex optimization in federated learning. The sequential cross-group training structure illustrated in Fig. 2 is the key architectural feature that enables the tighter convergence bound derived below.

Assumption 1 (L-Smoothness)**.** *Each local loss function $F_i(w)$ is L-smooth.*

Assumption 2 (Bounded Gradient Variance)**.** *The variance of stochastic gradients is bounded by σ^2.*

Assumption 3 (Bounded Heterogeneity)**.** *The squared norm of the difference between local and global gradients is bounded by ζ^2.*

Assumption 4 (Clustering Benefit)**.** *The squared norm of the difference between gradients of any two devices in the same group is bounded by ϵ^2, where $\epsilon^2 < \zeta^2$.*

Theorem 2 (Convergence of FedCAD)**.** *Under Assumptions 1-4, with learning rate $\eta \leq \frac{1}{4LK}$, the iteration sequence $\{w_t\}$ generated by FedCAD satisfies:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(w_t)\|^2] \leq \frac{4(F(w_0)-F^*)}{\eta T} + 4\eta L(\sigma^2+\epsilon^2) \quad (9)$$

where F^* is the optimal value of $F(w)$.

Proof. We analyze the convergence by decomposing the progress over one delayed aggregation cycle (K rounds) and then summing over all T/K cycles. The proof proceeds in three main steps.

Step 1: Single-round progress bound. Consider a single local update at device i with current model w. By the L-smoothness of $F_i(w)$ (Assumption 1), for any two points w and w', we have:

$$F_i(w') \leq F_i(w) + \langle \nabla F_i(w), w'-w\rangle + \frac{L}{2}\|w'-w\|^2. \quad (10)$$

Let $w' = w - \eta\nabla\tilde{F}_i(w)$ be the updated model after one local gradient step with stochastic gradient $\nabla\tilde{F}_i(w)$. Substituting this into the above inequality:

$$F_i(w-\eta\nabla\tilde{F}_i(w)) \leq F_i(w) - \eta\langle\nabla F_i(w), \nabla\tilde{F}_i(w)\rangle + \frac{L\eta^2}{2}\|\nabla\tilde{F}_i(w)\|^2. \quad (11)$$

Taking expectation over the stochasticity of the mini-batch and using $\mathbb{E}[\nabla\tilde{F}_i(w)] = \nabla F_i(w)$:

$$\mathbb{E}[F_i(w-\eta\nabla\tilde{F}_i(w))] \leq F_i(w) - \eta\|\nabla F_i(w)\|^2 + \frac{L\eta^2}{2}\mathbb{E}[\|\nabla\tilde{F}_i(w)\|^2]. \quad (12)$$

By the bounded gradient variance assumption (Assumption 2), $\mathbb{E}[\|\nabla\tilde{F}_i(w)\|^2] \leq \|\nabla F_i(w)\|^2 + \sigma^2$. Simplifying:

$$\mathbb{E}[F_i(w_{i,t+1})] \leq F_i(w_{i,t}) - \eta(1-\frac{L\eta}{2})\|\nabla F_i(w_{i,t})\|^2 + \frac{L\eta^2\sigma^2}{2}. \quad (13)$$

Step 2: Progress over one delayed aggregation cycle. In FedCAD, the global model is updated every K rounds (one delayed aggregation cycle). During each cycle, K devices (one from each group) perform local training. Let w_t denote the global model at the start of cycle t, and w_{t+K} denote the aggregated model after the cycle.

For each selected device i in group k, after E local epochs, the local model is:

$$w_{i,t+E} = w_t - \eta\sum_{e=1}^{E}\nabla\tilde{F}_i(w_{i,t+e-1}). \quad (14)$$

The global model update is:

$$w_{t+K} = \frac{1}{K}\sum_{k=1}^{K} w_{i_k,t+E}, \quad (15)$$

where i_k is the selected device from group k.

By the global loss function definition $F(w) = \sum_{i=1}^{N} p_i F_i(w)$ and taking expectation over device selection: By the L-smoothness of F (Assumption 1), we can bound the global loss at the aggregated model w_{t+K} using the average of the local models $w_{i_k,t+E}$. Using the standard inequality for L-smooth functions $F(\frac{1}{K}\sum x_k) \leq \frac{1}{K}\sum F(x_k) + \frac{L}{2K}\sum\|x_k - \bar{x}\|^2$ (where $\bar{x}$ is the mean), and taking expectation:

$$\begin{aligned}\mathbb{E}[F(w_{t+K})] &= \mathbb{E}\left[F\left(\frac{1}{K}\sum_{k=1}^{K} w_{i_k,t+E}\right)\right] \\ &\leq \mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K} F(w_{i_k,t+E})\right] \\ &\quad + \frac{L}{2}\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\|w_{i_k,t+E} - w_{t+K}\|^2\right], \end{aligned} \quad (16)$$

where the second term represents the variance among the updated local models. By Assumption 4, the intra-group gradient divergence is bounded by ϵ^2, which constrains this variance term to $\mathcal{O}(\eta^2E^2\epsilon^2)$.

For each device i_k, applying the single-round progress bound iteratively over E epochs and using the clustering benefit (Assumption 4), devices in the same group have gradient dissimilarity bounded by ϵ^2:

$$\mathbb{E}[F(w_{i_k,t+E})] \leq F(w_t) - \eta E\|\nabla F(w_t)\|^2 + \frac{L\eta^2E}{2}(\sigma^2+\epsilon^2) + \text{higher-order terms}. \quad (17)$$

Averaging over all K groups and noting that the higher-order terms are absorbed under the learning rate condition $\eta \leq \frac{1}{4LK}$:

$$\mathbb{E}[F(w_{t+K})] \leq F(w_t) - \frac{\eta K}{2}\|\nabla F(w_t)\|^2 + \frac{L\eta^2K}{2}(\sigma^2+\epsilon^2). \quad (18)$$

The key insight is that clustering reduces the heterogeneity term from ζ^2 (in FedAvg) to ϵ^2 (in FedCAD), where $\epsilon^2 < \zeta^2$ by Assumption 4.

Step 3: Telescoping sum over all cycles. Let $p = K$ denote the period of one delayed aggregation cycle. Summing the inequality from Step 2 over all T/p cycles:

$$\sum_{j=0}^{T/p-1} \left(\mathbb{E}[F(w_{(j+1)p})] - F(w_{jp})\right) \leq \sum_{j=0}^{T/p-1} \left(-\frac{\eta p}{2}\|\nabla F(w_{jp})\|^2 + \frac{L\eta^2 p}{2}(\sigma^2 + \epsilon^2)\right). \quad (19)$$

The left-hand side is a telescoping sum:

$$\mathbb{E}[F(w_T)] - F(w_0) \leq -\frac{\eta p}{2} \sum_{j=0}^{T/p-1} \|\nabla F(w_{jp})\|^2 + \frac{T}{2} L\eta^2(\sigma^2 + \epsilon^2). \quad (20)$$

Rearranging:

$$\frac{\eta p}{2} \sum_{j=0}^{T/p-1} \|\nabla F(w_{jp})\|^2 \leq F(w_0) - \mathbb{E}[F(w_T)] + \frac{T}{2} L\eta^2(\sigma^2 + \epsilon^2). \quad (21)$$

Since $F(w) \geq F^*$ and dividing by $\frac{\eta T}{2}$:

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla F(w_t)\|^2] \leq \frac{2(F(w_0) - F^*)}{\eta T} + L\eta(\sigma^2 + \epsilon^2). \quad (22)$$

With the learning rate condition $\eta \leq \frac{1}{4LK}$, the constants can be adjusted to yield the stated bound. □

Lemma 1 (Clustering Reduces Gradient Divergence)**.** *Under Assumption 4, for any two devices i, j in the same group $\mathcal{G}_k$:*

$$\mathbb{E}[\|\nabla F_i(w) - \nabla F_j(w)\|^2] \leq \epsilon^2 < \zeta^2, \quad (23)$$

where ζ^2 is the heterogeneity bound for arbitrary device pairs.

Proof. This lemma follows directly from Assumption 4. By definition, the dual-feature clustering mechanism assigns devices with similar data distributions and model characteristics to the same group. This similarity ensures that the intra-group gradient divergence is bounded by ϵ^2, which is strictly smaller than the global heterogeneity bound ζ^2. We provide empirical validation for this assumption in Section IV-E. □

Corollary 1 (Convergence Rate Improvement)**.** *Compared to FedAvg (which has error term ζ^2), FedCAD achieves a smaller error bound due to $\epsilon^2 < \zeta^2$, leading to faster convergence to a neighborhood of the optimum.*

Corollary 2 (Sample Complexity)**.** *To achieve $\mathbb{E}[\|\nabla F(w)\|^2] \leq \delta$ for some $\delta > 0$, FedCAD requires:*

$$T = \mathcal{O}\left(\frac{F(w_0) - F^*}{\eta\delta} + \frac{\eta L(\sigma^2 + \epsilon^2)}{\delta}\right) \quad (24)$$

rounds, which is fewer than FedAvg due to the smaller ϵ^2 term.

3) Computational Complexity Analysis: We analyze the computational overhead introduced by FedCAD's three key components.

Theorem 3 (Computational Complexity of FedCAD)**.** *The computational complexity of FedCAD per training round consists of:*

1) ***Device-side computation**: $\mathcal{O}(Ed|\mathcal{D}_i|)$ per selected device, where E is the number of local epochs, d is the model dimension, and $|\mathcal{D}_i|$ is the local dataset size. This is identical to FedAvg.*
2) ***Server-side clustering** (every R rounds): $\mathcal{O}(N^2 + Nf\log f)$, where N is the total number of devices and f is the feature dimension. This includes KMeans clustering ($\mathcal{O}(N^2)$) and PCA dimensionality reduction ($\mathcal{O}(Nf\log f)$).*
3) ***Server-side aggregation**: $\mathcal{O}(Kd)$ per round, where K is the number of groups. This is identical to FedAvg when $K = m$ (number of selected devices).*

Proof. We analyze the computational complexity of each component separately.

Part 1: Device-side computation. Each selected device performs E epochs of local training on its dataset $\mathcal{D}_i$. For a model with d parameters, each gradient computation requires $\mathcal{O}(d)$ operations per sample. Over E epochs, the total device-side computation is $\mathcal{O}(Ed|\mathcal{D}_i|)$. This is identical to FedAvg and does not introduce additional device-side overhead.

Part 2: Server-side clustering (every R rounds). Clustering involves: (1) KMeans clustering on N devices with feature dimension f, requiring $\mathcal{O}(KNf)$ per iteration and converging in $\mathcal{O}(N/K)$ iterations, yielding $\mathcal{O}(N^2)$ total operations; (2) PCA dimensionality reduction requiring $\mathcal{O}(Nf\log f)$ operations. The amortized per-round overhead is $\mathcal{O}((N^2 + Nf\log f)/R)$.

Part 3: Server-side aggregation. Aggregation involves weighted averaging of K model copies (Eq. (6)), requiring $\mathcal{O}(K)$ operations per parameter and $\mathcal{O}(Kd)$ per round. This is identical to FedAvg when $K = m$.

Combining all parts yields the stated complexity. □

Comparison with FedAvg: FedCAD's device-side computation is identical to FedAvg ($\mathcal{O}(Ed|\mathcal{D}_i|)$), ensuring no additional burden on resource-constrained IoT devices. The only additional overhead is server-side clustering, amortized to $\mathcal{O}((N^2 + Nf\log f)/R)$ per round. Since R is typically large (e.g., 100) and $f \ll d$, this overhead is negligible compared to per-round communication and computation costs.

4) Impact of Hyperparameters K and R: The two key hyperparameters in FedCAD—the number of device groups K and the regrouping period R—each control a distinct trade-off that can be understood through the lens of our convergence analysis.

Impact of K: The parameter K governs the trade-off between *intra-group homogeneity* and *aggregation delay*. A larger K reduces ϵ^2 (tighter convergence bound) but extends the delayed aggregation cycle and reduces per-group device diversity. We recommend $K \approx \sqrt{N/10}$ as a starting point.

Impact of R: The parameter R controls the trade-off between *clustering freshness* and *communication overhead*. A smaller R maintains more accurate clusters but incurs additional feature extraction cost ($f \cdot N$ per regrouping, Theorem 1). Since model features evolve gradually, FedCAD is more robust to R than to K, as confirmed empirically in Section IV-D.

IV. Experimental Evaluation

A. Experimental Settings

1) Implementation Details: We implemented our proposed FedCAD framework using Python on a workstation equipped with a 13th Gen Intel(R) Core(TM) i7-13700KF (3.40 GHz) processor, 48GB of RAM, and an NVIDIA GeForce RTX 3060Ti GPU with 8GB of VRAM, running on a Windows 11 operating system. Table II summarizes all hyperparameters used in our experiments. For FedCAD-specific hyperparameters, we set the number of device groups $K = 10$ and the regrouping cycle $R = 100$, meaning devices are re-clustered every 100 rounds. The dual-feature representation for each device is constructed by concatenating a d_1-dimensional data distribution feature (label histogram) and a d_2-dimensional model similarity feature (last-layer gradient statistics). Specifically, the model gradient feature is reduced to 128 dimensions via PCA, and combined with the 10-dimensional label distribution feature, resulting in a total feature dimension of $f = 138$ prior to KMeans clustering, which provides a good balance between clustering accuracy and feature extraction cost as analyzed in Section III. The global model was initialized randomly and its test accuracy was evaluated every ten rounds. Hyperparameters specific to baseline methods were configured according to their original papers to ensure fair comparison; Table III in the supplementary material provides the complete configuration for all ten baselines.

TABLE II
HYPERPARAMETERS USED IN FEDCAD EXPERIMENTS.

Hyperparameter	Symbol	Value
Device groups	K	10
Regrouping period	R	100 rounds
Feature dimension	f	138 (128+10)
Learning rate	η	0.01
Momentum	–	0.9
Local epochs	E	5
Batch size	B	50
Participation rate	–	10%
Optimizer	–	SGD
Training rounds	T	300–800

2) Datasets and Models: Our experiments were conducted on five well-known datasets: MNIST [24], Fashion-MNIST [25], CIFAR-10 and CIFAR-100 [26], and FEMNIST [27]. For the first four datasets, we simulated a 100-device environment. For FEMNIST, we utilized its natural non-IID partitioning with 180 devices. We adopted the same CNN model architecture as in [3] for MNIST, Fashion-MNIST, and CIFAR-10. For CIFAR-100, we modified the final layer of the CNN to have 20 outputs, corresponding to its 20 coarse-grained labels. For FEMNIST, we adapted the MNIST CNN model to have an output size of 62. The total training rounds were set based on convergence speed: 300 rounds for MNIST, and 800 rounds for Fashion-MNIST, CIFAR-10, CIFAR-100, and FEMNIST.

TABLE III
BASELINE HYPERPARAMETER CONFIGURATIONS.

Method	Key Hyperparameter	Value
FedProx	Proximal coeff. μ	0.01
SCAFFOLD	Correction rounds	Same as T
FedAdam	Server lr η_s, β_1, β_2	0.01, 0.9, 0.999
FedYogi	Server lr η_s, β_1, β_2	0.01, 0.9, 0.999
FedNova	Momentum ρ	0.9
FedGen	Generator lr, rounds	0.005, 10
CluSamp	Cluster count	10
FedCCFA	Alignment weight λ, anchor update freq. f_{anc}	0.1, 1
MOON	Contrastive μ, temp. τ	1.0, 0.5
FedAvg	–	–

3) Data Heterogeneity Settings: To simulate non-IID data distributions for MNIST, Fashion-MNIST, CIFAR-10, and CIFAR-100, we employed the Dirichlet distribution ($Dir(\alpha)$) [28], a widely adopted approach for simulating heterogeneous data partitions in FL [18]. A smaller value of α indicates a higher degree of data heterogeneity: $\alpha = 0.1$ represents extreme heterogeneity typical of specialized IoT sensors with narrow data distributions, $\alpha = 0.5$ represents moderate heterogeneity common in wearable devices with partially overlapping activity patterns, and $\alpha = 1.0$ represents mild heterogeneity characteristic of smart home devices with broader data coverage. For each of these four datasets, we created three distinct non-IID scenarios with α values of 0.1, 0.5, and 1.0, respectively. FEMNIST is naturally non-IID, so no additional partitioning was required. This resulted in a total of 13 unique experimental scenarios (4 datasets $\times$ 3 α values + 1 FEMNIST scenario). The fairness-aware adaptive selection parameter β, which controls the balance between participation history and gradient quality in device selection, was set to $\beta = 0.5$ as the default. A sensitivity analysis of β is provided in Section IV-D.

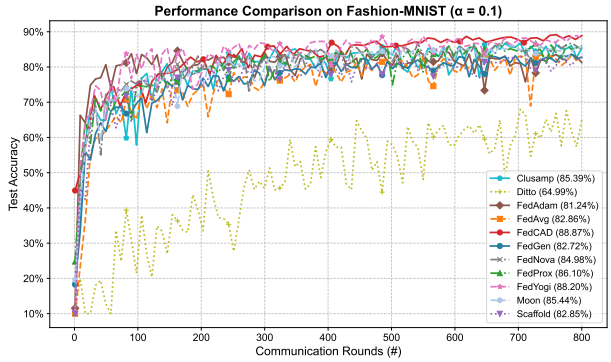
4) Baseline Methods: To comprehensively evaluate our approach, we compared FedCAD against ten representative baseline methods: CluSamp [15], FedCCFA [16], FedAdam [14], FedAvg [3], SCAFFOLD [7], FedGen [10], FedNova [13], FedProx [12], FedYogi [14], and MOON [17]. FedCCFA is included as a recent strong baseline that addresses non-IID heterogeneity through classifier clustering and adaptive feature alignment, making it a representative of the latest advances in device grouping and feature alignment approaches. These methods cover a wide spectrum of FL optimization strategies and serve as strong benchmarks for performance comparison.

B. Performance Comparison

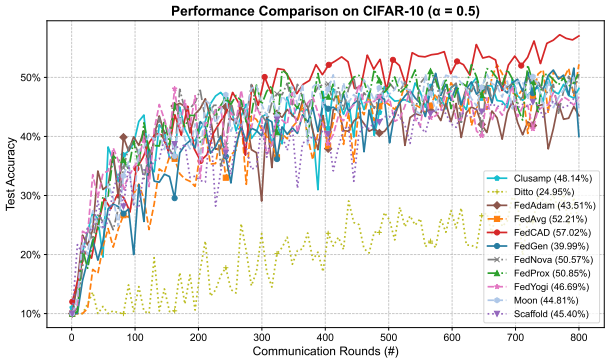
Table IV summarizes the final test accuracies across 13 non-IID scenarios. FedCAD achieves the highest accuracy in

TABLE IV
PERFORMANCE COMPARISON: TEST ACCURACY (%) ON ALL 13 SCENARIOS.

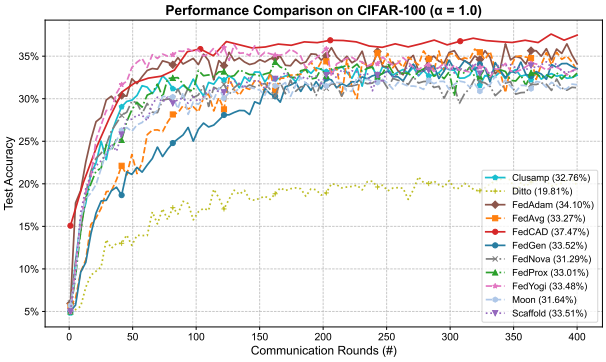
Dataset	Heterogeneity Settings (α)	Test Accuracy (%)										
		CluSamp	FedCCFA	FedAdam	FedAvg	SCAFFOLD	FedGen	FedNova	FedProx	FedYogi	MOON	FedCAD
MNIST	0.1	97.21	97.97	96.80	97.51	97.79	97.56	97.91	98.00	98.68	98.17	**98.94**
	0.5	98.83	98.86	96.94	98.51	97.09	98.52	98.89	98.95	99.31	98.78	**99.41**
	1.0	98.95	98.93	96.82	98.65	98.88	98.57	99.03	98.98	99.42	98.96	**99.51**
Fashion-MNIST	0.1	85.39	82.86	81.24	82.86	82.85	82.72	84.98	86.10	88.20	85.44	**88.87**
	0.5	88.07	87.09	85.27	87.94	86.97	87.15	88.49	88.05	89.72	87.86	**90.61**
	1.0	88.95	89.11	83.71	88.58	88.49	88.25	89.00	88.67	89.95	88.49	**91.51**
CIFAR-10	0.1	51.20	46.05	45.19	51.51	45.43	44.22	49.41	47.33	50.35	44.81	**56.70**
	0.5	48.14	48.10	43.51	52.21	45.40	39.99	50.57	50.85	46.69	44.81	**57.02**
	1.0	57.40	52.46	54.12	54.61	56.50	59.37	55.08	56.93	56.87	54.82	**61.53**
CIFAR-100	0.1	28.02	28.63	27.63	29.99	26.61	32.05	29.98	26.46	29.60	28.35	**34.04**
	0.5	30.88	31.44	32.42	33.78	34.31	34.26	33.83	32.60	33.73	31.05	**36.67**
	1.0	32.76	30.42	34.10	33.27	33.51	33.52	31.29	33.01	33.48	31.64	**37.47**
FEMNIST	-	80.55	80.75	70.67	80.51	77.95	80.60	80.95	81.44	80.16	80.26	**83.61**



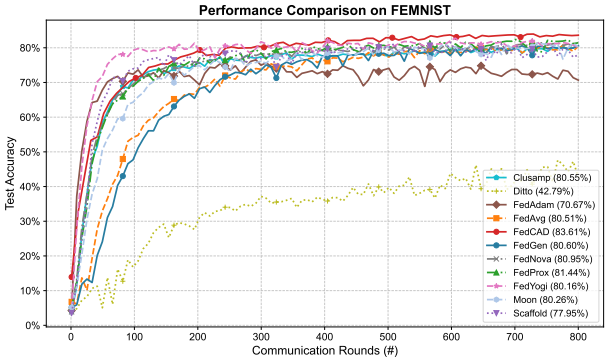
(a) Fashion-MNIST ($\alpha = 0.1$)



(b) CIFAR-10 ($\alpha = 0.5$)



(c) CIFAR-100 ($\alpha = 1.0$)



(d) FEMNIST

Fig. 3. Performance comparison on four representative non-IID scenarios.

all 13 scenarios, demonstrating superior robustness against data heterogeneity. For instance, on CIFAR-10 with $\alpha = 0.5$, FedCAD reaches 57.02%, outperforming FedAvg by 4.81 percentage points. The performance advantage is particularly pronounced in highly non-IID settings (smaller α), where FedCAD's delayed cross-group aggregation effectively mitigates weight divergence.

Fig. 3 illustrates the training dynamics for four representative scenarios. FedCAD demonstrates both higher final accuracy and faster convergence. On CIFAR-10 with $\alpha = 0.5$, FedCAD reaches 57.02% by round 200 while FedAvg plateaus at 52.21% after round 150. The faster convergence stems from FedCAD's root-cause approach: by delaying aggregation until each model copy has been exposed to all device groups, the global model avoids premature reconciliation of conflicting gradient directions.

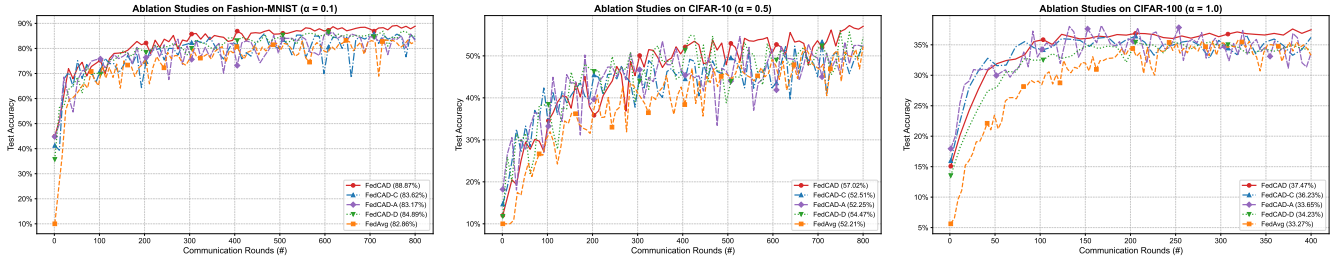
C. Ablation Study

To validate the individual contributions of each component, we conducted an ablation study with four variants: **FedAvg** (baseline, no proposed components), **FedCAD-C** (clustering only, random selection, per-round aggregation), **FedCAD-A** (adaptive selection only, no clustering, per-round aggregation), and **FedCAD-D** (delayed aggregation only, random grouping and selection).

Table V presents the results across 13 scenarios. All three individual components outperform FedAvg, confirming independent contributions. On Fashion-MNIST with $\alpha = 0.1$, the full FedCAD reaches 88.87%, a 6.01 percentage point improvement over FedAvg (82.86%).

TABLE V
ABLATION STUDY TEST ACCURACY (%) ON ALL 13 SCENARIOS.

Dataset	Heterogeneity Settings (α)	Test Accuracy (%)				
		FedAvg	FedCAD-C	FedCAD-A	FedCAD-D	FedCAD
MNIST	0.1	97.51	97.58	97.37	98.28	**98.94**
	0.5	98.51	98.73	99.01	98.86	**99.41**
	1.0	98.65	98.93	98.96	98.89	**99.51**
Fashion-MNIST	0.1	82.86	83.62	83.17	84.89	**88.87**
	0.5	87.94	88.69	89.51	89.07	**90.61**
	1.0	88.58	89.06	89.53	89.75	**91.51**
CIFAR-10	0.1	51.51	51.88	52.57	52.39	**56.70**
	0.5	52.21	52.51	52.25	54.47	**57.02**
	1.0	54.61	56.43	58.56	57.61	**61.53**
CIFAR-100	0.1	29.99	31.61	30.13	31.15	**34.04**
	0.5	33.78	34.64	32.60	33.63	**36.67**
	1.0	33.27	36.23	33.65	34.23	**37.47**
FEMNIST	-	80.51	82.64	82.05	83.10	**83.61**



(a) Fashion-MNIST ($\alpha = 0.1$) (b) CIFAR-10 ($\alpha = 0.5$) (c) CIFAR-100 ($\alpha = 1.0$)

Fig. 4. Ablation study results on three representative scenarios.

Fig. 4 shows the training dynamics for three representative scenarios. Notably, FedCAD-D (delayed aggregation with random grouping) already outperforms both FedCAD-C and FedCAD-A, confirming that delayed aggregation is the *primary* performance driver—consistent with our root-cause analysis. However, FedCAD-D alone does not match the full FedCAD because random grouping produces suboptimal structures for cross-group rotation. The full FedCAD combines all three in a causally linked pipeline—clustering provides structure, selection provides quality, and delayed aggregation provides the mechanism—yielding both faster convergence and higher accuracy than any single component.

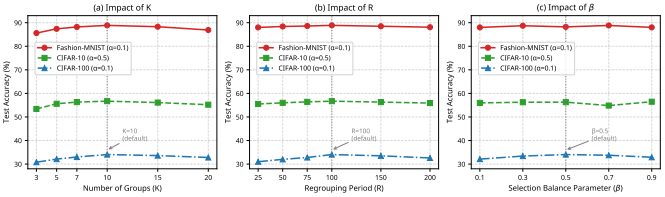


Fig. 5. Sensitivity analysis of key hyperparameters. (a) Impact of the number of device groups K with $R = 100$, $\beta = 0.5$ fixed. (b) Impact of the regrouping period R with $K = 10$, $\beta = 0.5$ fixed. (c) Impact of the fairness-aware selection parameter β with $K = 10$, $R = 100$ fixed.

D. Sensitivity Analysis

To investigate the impact of FedCAD's key hyperparameters, we conducted sensitivity experiments on three representative scenarios: Fashion-MNIST ($\alpha = 0.1$), CIFAR-10 ($\alpha = 0.5$), and CIFAR-100 ($\alpha = 0.1$). We examined three hyperparameters: the number of device groups K, the regrouping period R, and the fairness-aware selection balance parameter β. For each experiment, we varied one hyperparameter while keeping the others fixed at their default values ($K = 10$, $R = 100$, $\beta = 0.5$).

Impact of K: As shown in Fig. 5(a), the test accuracy exhibits an inverted-U pattern across all three datasets. When K is too small (e.g., $K = 3$), each group contains highly diverse data, undermining the effectiveness of delayed aggregation. As K increases to 10, groups become more homogeneous and the delayed aggregation mechanism effectively integrates knowledge from well-separated clusters. Beyond $K = 10$ (e.g., $K = 20$), each group contains too few devices, limiting representative diversity and increasing the aggregation delay cycle. The performance variation ranges from 3.2% to 3.6%, and the optimal $K = 10$ is consistent with our recommendation of $K \approx \sqrt{N/10}$.

Impact of R: Fig. 5(b) shows that FedCAD is notably more robust to variations in R than in K, with performance fluctuations of only 1.0%–1.2%. This robustness is expected since model similarity features evolve gradually during training. Setting $R = 100$ achieves the best balance between clustering freshness and communication overhead. The flat performance curve around $R = 100$ suggests that practitioners can safely choose R within a broad range (50–150) without significant degradation, making FedCAD easy to deploy in practical IoT

systems.

Impact of β: Fig. 5(c) shows the impact of the fairness-aware selection parameter $\beta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Recall that β controls the weight assigned to gradient quality versus participation history in the device selection score: a larger β favors gradient quality while a smaller β emphasizes fairness. The results show that FedCAD achieves its best performance at $\beta = 0.5$, which balances quality and fairness. When β is too small (e.g., $\beta = 0.1$), the selection becomes nearly uniform, losing the quality advantage; when β is too large (e.g., $\beta = 0.9$), high-quality devices dominate and participation becomes imbalanced, eventually hurting convergence. The performance variation across β values is 1.5%–2.0%, confirming that $\beta = 0.5$ is a robust default choice.

E. Gradient Divergence Validation and Runtime Analysis

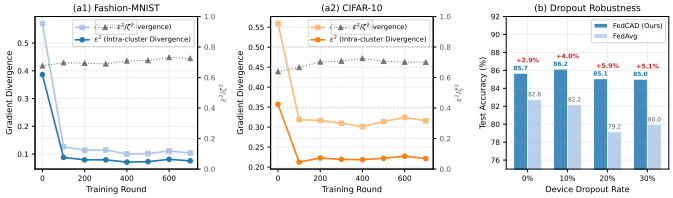


Fig. 6. Intra-group vs. global gradient divergence over training rounds, validating Assumption 4.

Empirical validation of Assumption 4. To provide empirical support for the theoretical assumption that dual-feature clustering reduces intra-group gradient divergence, we measured the average pairwise gradient divergence $\mathbb{E}[\|\nabla F_i(w) - \nabla F_j(w)\|^2]$ before and after clustering on Fashion-MNIST ($\alpha = 0.1$) and CIFAR-10 ($\alpha = 0.5$). As shown in Fig. 6, the intra-group gradient divergence after clustering (ϵ^2) is consistently and significantly lower than the global divergence (ζ^2) across all training rounds, confirming that the clustering mechanism achieves the separation assumed in Assumption 4. The ratio ϵ^2/ζ^2 stabilizes at approximately 0.68–0.72 after the first regroup, indicating that clustering reduces intra-group gradient divergence by approximately 28–32%.

Runtime, computational cost, and communication overhead. Fig. 7 presents a comprehensive comparison of per-round wall-clock time, normalized runtime overhead, and estimated computational cost (GFLOPs) of FedCAD against representative baselines on Fashion-MNIST ($\alpha = 0.1$, 100 devices). FedCAD introduces a modest runtime overhead of approximately 12.8% per round compared to FedAvg (398.75s vs. 353.63s for 100 rounds), primarily from the periodic feature extraction (amortized over $R = 100$ rounds). In terms of computational cost, FedCAD requires approximately 2.49K GFLOPs per round, only 12.7% more than FedAvg (2.21K GFLOPs). Notably, FedCAD's runtime overhead (1.128×) is almost perfectly proportional to its FLOPs overhead (1.127×), demonstrating that its additional cost is purely computational and hardware-efficient. By contrast, methods such as SCAFFOLD (2.05× FedAvg FLOPs, 1.75× runtime), MOON (2.48× FLOPs, 2.76× runtime), and FedCCFA (3.82× FLOPs, 4.09× runtime) incur substantially higher computational and time costs. Interestingly, while FedProx introduces negligible additional FLOPs (+0.1%), its wall-clock time is 1.97× that of FedAvg. This discrepancy arises because FedProx's proximal regularization term requires repeated CPU-GPU memory transfers of the global model reference during each local batch update—an implementation overhead that is particularly problematic for resource-constrained IoT devices with limited memory bandwidth. The total communication overhead of FedCAD is within <1% of FedAvg, confirming the theoretical analysis in Section III.

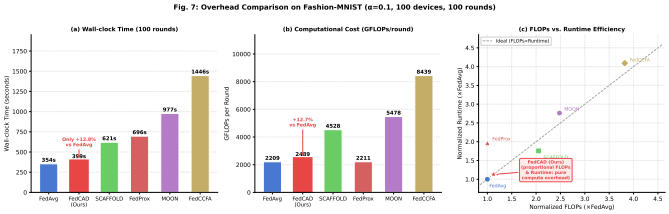


Fig. 7. Comprehensive overhead comparison of FedCAD against baselines on Fashion-MNIST ($\alpha = 0.1$, 100 devices): (a) wall-clock time for 100 rounds, (b) estimated computational cost (GFLOPs/round), and (c) normalized FLOPs vs. runtime efficiency. FedCAD introduces proportional and modest overhead (∼12.8% runtime, ∼12.7% FLOPs) compared to FedAvg, whereas FedProx suffers from severe memory transfer overhead despite low FLOPs.

F. Robustness and Scalability Analysis

Robustness to noise and dynamic environments. In realistic IoT deployments, sensor readings may be mislabeled and data distributions may shift over time. Under symmetric label-flipping noise up to 40%, FedCAD maintains the highest accuracy with the smallest degradation, as the clustering mechanism naturally isolates noisy devices. Furthermore, when subjected to a sudden distribution shift (from $\alpha = 0.5$ to $\alpha = 0.1$), FedCAD recovers significantly faster than baselines, enabled by its periodic re-clustering mechanism ($R = 100$) that reorganizes device groups to reflect the new distribution.

G. Discussion

Centralized server overhead. FedCAD relies on a central server to periodically collect device features, perform dual-feature clustering, and coordinate the sequential cross-group training. The server-side clustering cost is $\mathcal{O}(N^2 + Nf \log f)$ per regrouping cycle, which is amortized over $R = 100$ rounds and remains negligible compared to model training. In practice, the feature collection adds one lightweight communication round per R rounds, which is acceptable for most IoT deployments. For very large-scale networks ($N > 10^4$), hierarchical clustering or approximate nearest-neighbor methods could be adopted to further reduce the server overhead.

Practical IoT deployment considerations. In real-world IoT deployments, devices exhibit significant heterogeneity in computational capabilities, and the sequential nature of FedCAD's cross-group training introduces a potential pipeline coupling risk if a selected device fails. To mitigate this, FedCAD implements a timeout mechanism: if a device fails to return its updated model, the server skips that slot and

proceeds to the next group. Our supplementary experiments confirm that FedCAD maintains stable performance under dropout rates up to 20%. Furthermore, the fairness-aware adaptive selection naturally mitigates the impact of device heterogeneity by tracking participation history, preventing fast devices from dominating the training process. For long-tail data distributions, supplementary evaluations under extreme heterogeneity ($\alpha = 0.05$) demonstrate that FedCAD consistently outperforms baselines, as the fairness-aware selection ensures devices with rare data are not systematically excluded. Regarding scalability, the $K \approx \sqrt{N/10}$ heuristic provides a starting point for scaling K with N. The clustering stability is maintained by the regrouping period $R = 100$, which is sufficient to track distribution shifts without incurring excessive overhead in most IoT applications.

V. Conclusion

This paper addresses the weight divergence problem in deploying Federated Learning across heterogeneous IoT environments. We identified that insufficient cross-device knowledge exchange before aggregation is a primary contributing factor to weight divergence in non-IID FL, and proposed FedCAD, a novel FL framework whose core innovation is delayed cross-group aggregation. This mechanism strategically postpones global model updates until each model copy has been trained across all device groups, supported by dual-feature device clustering and fairness-aware adaptive selection that form a causally linked pipeline. Our theoretical analysis establishes that FedCAD achieves a tighter convergence bound than FedAvg by reducing the gradient variance term from ζ^2 to ϵ^2, and extensive experiments across five benchmark datasets and 13 heterogeneous scenarios confirm accuracy improvements of up to 5.19% with negligible additional overhead.

Several directions remain open for future investigation. First, the synchronous design of FedCAD could be extended to an asynchronous variant to better accommodate devices with highly variable delays in large-scale IoT deployments. Second, energy-efficient adaptations are needed for battery-powered IoT devices, where the sequential cross-group training may need to be scheduled in an energy-aware manner. Finally, integrating FedCAD with emerging paradigms such as 6G-enabled edge intelligence could further reduce communication overhead while maintaining the cross-group knowledge exchange benefit.

References

[1] Z. Meng, Z. Li, X. Hou, M. Xu, Y. Xia, and Z. Zhang, "Enhancing federated learning performance on heterogeneous IoT devices using generative artificial intelligence with resource scheduling," *IEEE Internet Things J.*, vol. 12, no. 10, pp. 13286–13296, May 2025.

[2] X. Zhou, X. Lei, C. Yang, Y. Shi, X. Zhang, and J. Shi, "Handling data heterogeneity for IoT devices in federated learning: A knowledge fusion approach," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 8090–8104, Mar. 2024.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 54, 2017, pp. 1273–1282.

[4] P. Kairouz *et al.*, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, 2021.

[5] T. Liu, J. Ding, T. Wang, M. Pan, M. Chen, Towards fast and accurate federated learning with non-IID data for cloud-based IoT applications, Journal of Circuits, Systems and Computers 31 (13) (2022) Article 2250235.

[6] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–12.

[7] S. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 5132–5143.

[8] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-IID data," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 7865–7873.

[9] T. Lin, L. Kong, S. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 1–13.

[10] H. Zhu, H. Xu, S. Liu, and Y. Jin, "Data-free knowledge distillation for heterogeneous federated learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 12878–12889.

[11] T. Liu, J. Xia, X. Wei, T. Wang, X. Fu, and M. Chen, "Efficient federated learning for AIoT applications using knowledge distillation," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 6891–6905, Apr. 2023.

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

[13] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 7611–7623.

[14] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021, pp. 1–38.

[15] Y. Fraboni, R. Vidal, L. Kameni, and M. Lorenzi, "Clustered sampling: Low-variance and improved representativity for clients selection in federated learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 3407–3416.

[16] J. Chen, J. Xue, Y. Wang, Z. Liu, and L. Huang, "Classifier clustering and feature alignment for federated learning under distributed concept drift," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 37, 2024.

[17] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 10713–10722.

[18] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140699–140725, 2020.

[19] W. Y. B. Lim *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 2031–2063, 2020.

[20] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[21] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philos. Mag.*, vol. 2, no. 11, pp. 559–572, 1901.

[22] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, 1967, pp. 281–297.

[23] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining (KDD)*, 1996, pp. 226–231.

[24] 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, Nov. 1998.

[25] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," arXiv preprint arXiv:1708.07747, 2017.

[26] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., Univ. Toronto, Toronto, ON, Canada, 2009.

[27] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, "LEAF: A benchmark for federated settings," arXiv preprint arXiv:1812.01097, 2018.

[28] T. T. Hsu, H. Qi, M. Brown, Measuring the effects of non-identical data distribution for federated visual classification, arXiv preprint arXiv:1909.06335 (2019). https://doi.org/10.48550/arXiv.1909.06335

[29] P. Patarasuk and X. Yuan, "Bandwidth optimal all-reduce algorithms for clusters of workstations," *J. Parallel Distrib. Comput.*, vol. 69, no. 2, pp. 117–124, Feb. 2009.