FedCAP: Robust Federated Learning via Customized Aggregation and Personalization

Youpeng Li[1], Xinda Wang[1], Fuxun Yu[2], Lichao Sun[3], Wenbin Zhang[4], Xuyu Wang[4*]
[1]University of Texas at Dallas, Richardson, United States
[2]Microsoft, Redmond, United States
[3]Lehigh University, Bethlehem, United States
[4]Florida International University, Miami, United States
{youpeng.li,xinda.wang}@utdallas.edu, fuxunyu@microsoft.com,
lis221@lehigh.edu, {wenbin.zhang, xuywang}@fiu.edu

***Abstract*—Federated learning (FL), an emerging distributed machine learning paradigm, has been applied to various privacy-preserving scenarios. However, due to its distributed nature, FL faces two key issues: the non-independent and identical distribution (non-IID) of user data and vulnerability to Byzantine threats. To address these challenges, in this paper, we propose FedCAP, a robust FL framework against both data heterogeneity and Byzantine attacks. The core of FedCAP is a model update calibration mechanism to help a server capture the differences in the direction and magnitude of model updates among clients. Furthermore, we design a customized model aggregation rule that facilitates collaborative training among similar clients while accelerating the model deterioration of malicious clients. With a Euclidean norm-based anomaly detection mechanism, the server can quickly identify and permanently remove malicious clients. Moreover, the impact of data heterogeneity and Byzantine attacks can be further mitigated through personalization on the client side. We conduct extensive experiments, comparing multiple state-of-the-art baselines, to demonstrate that FedCAP performs well in several non-IID settings and shows strong robustness under a series of poisoning attacks.**

***Index Terms*—federated learning, data heterogeneity, Byzantine-robustness**

I. Introduction

With the emergence of large foundation models [1], model performance increasingly relies on high-quality and high-volume data. In fields such as Internet of Things (IoT) [2]–[4] and healthcare [5], [6], user data often contains a large amount of sensitive information. Various privacy-preserving policies such as the General Data Protection Regulation (GDPR) [7], [8] restrict the collection of user data by a central server. As an emerging distributed machine learning paradigm, federated learning (FL) [9] allows user data to remain local while coordinating clients to train a global model. Due to its distributed nature, FL faces two key issues. First, statistical heterogeneity exists in user data. In real-world FL applications, such as Google's next word prediction, training a single global model that caters to the individual needs of all users is challenging due to their diverse language habits and regional cultures [10]. Second, FL systems are vulnerable to Byzantine threats [11], [12], with malicious clients uploading arbitrary model updates to the server, which can greatly degrade model performance on any test inputs (i.e., untargeted poisoning attack [13]).

[*]Corresponding author

To mitigate the impact of data heterogeneity, the concept of personalized FL is introduced [14], where each client holds a personalized model to fit its own data distribution better. However, most personalized FL algorithms [15]–[17] fail to adapt to non-independent and identically distributed (non-IID) settings. Therefore, we need to address **Challenge 1**:

- *How to design a personalized FL framework that exhibits adaptiveness in various heterogeneous data settings?*

To defend against poisoning attacks, existing robust FL methods adopt diverse strategies, with some focusing on the server side such as detection [18], [19] and robust aggregation [20]–[23], while others concentrate on the client side through personalization [15]. However, the above robust FL methods are less effective against attacks in non-IID settings [24], [25], due to the difficulties in distinguishing malicious behavior from clients. This leads to varying degrees of aggregation knowledge loss while defending against attacks, which in turn results in model performance degradation. Moreover, in settings with strong attacks [26]–[28], malicious clients can camouflage themselves as benign, making it more difficult for robust FL methods to detect them and causing further deterioration in the benign models. Therefore, it is imperative to tackle **Challenge 2**:

- *How can we design a Byzantine-robust FL framework that precisely distinguishes between benign and malicious clients in non-IID settings without causing a significant loss in model accuracy?*

For real-world applications, a unified FL framework considering both challenges is needed, but few studies focus on this. Although Ditto [15] mitigates the impact of data heterogeneity and attacks via personalization, one of its limitations is that it does not directly detect malicious clients and instead requires a trade-off between model utility and robustness. A naive strategy is to combine robust aggregation rules (AGRs) (e.g., Median, Trimmed Mean [20], and ClusteredFL [22]) with client personalization (e.g., Ditto), but their inherent limitations lead to combinations that fail to improve model performance (see empirical results in Fig. 8).

Motivated by the limitations of the above approaches, we propose FedCAP, a robust FL framework against both data heterogeneity and Byzantine attacks. FedCAP mainly includes four components. A model calibration mechanism helps distinguish malicious model updates from benign ones in non-IID settings. A customized aggregation rule can then facilitate collaboration among similar clients and accelerate malicious model deterioration. With the incorporation of an anomaly detection mechanism, the server is able to identify and permanently remove malicious clients. A personalized training module can further mitigate the impact of data heterogeneity and attacks, building on the relatively clean customized models. These components enable FedCAP to adapt to various non-IID settings and types of attacks.

FedCAP utilizes two key insights observed and analyzed through experiments in Section III-C. First, following the principle of ”good becomes better and bad becomes worse”, we can promote collaboration among similar clients in benign scenarios and inhibit cooperation between benign and malicious clients in attack scenarios. Second, we observe abnormal behavior from malicious clients when they intrude into FL training. Specifically, we observe that as the number of global rounds increases, the average Euclidean norm of the model updates from malicious clients gradually rises, thus degrading the model performance of benign clients.

Building upon the above insights, we design the customized aggregation rule to address **Challenge 1**. The server assigns each client a customized model that closely matches its data distribution, determined by the contributions from other clients. Specifically, we use the normalized cosine similarity of model updates among clients as aggregation weights to customize the models. By adjusting the scale factor of normalization, FedCAP achieves adaptation for various non-IID settings. To tackle **Challenge 2**, we propose the model update calibration mechanism and the anomaly detection mechanism based on the Euclidean norm of calibrated model updates. Specifically, by calibrating the uploaded model updates, FedCAP effectively captures differences in the magnitude and direction of model updates between benign and malicious clients. Combining this with the customized aggregation, FedCAP accelerates the model deterioration of malicious clients, leading to a significant increase in the Euclidean norm of their calibrated model updates. This triggers the anomaly detection mechanism, allowing the server to identify and remove malicious clients. Therefore, FedCAP excels in distinguishing between benign and malicious clients in non-IID settings.

Our main contributions are summarized as follows:

- We propose FedCAP, a robust FL framework against both data heterogeneity and attacks, which adapts to various non-IID settings and different types of attacks.
- We propose a model update calibration mechanism that excels in capturing the difference in the direction and magnitude of client model updates in non-IID settings.
- We design a customized model aggregation rule, which facilitates collaboration among similar clients while accelerating the model deterioration of malicious clients, helping the server identify and remove them permanently by triggering an anomaly detection mechanism based on the Euclidean norm of calibrated model updates.
- We perform extensive experiments indicating that the proposed FedCAP outperforms the state-of-the-art (SOTA) FL baselines in terms of model accuracy and robustness.

Open science. The source code and data artifacts of FedCAP have been open-sourced at Github.

II. Related Work

Federated Learning with Non-IID Data. FL can be broadly categorized into two types: single-model FL and multi-model FL. In single-model FL, clients collaboratively train a single global model [9]. Existing research primarily focuses on two key aspects: enhancing the generalizability of the global model [16], [29], [30] and developing personalized FL algorithms to mitigate the impact of data heterogeneity [15]–[17], [31]. For example, FedRoD [16] improves the generalizability of the global model by using balanced softmax loss to mitigate the effect of label distribution skew. FedAvg-FT [32] treats clients' updated local models as personalized models and evaluates their performance. Ditto [15] trains a personalized model for each client while locally updating the global model.

Given the limitation of generalizability of the single global model, the concept of multi-model FL, which refers to the server holding multiple models, is introduced. For example, in clustering-based FL algorithms [22], [33], the server divides clients into multiple clusters, and within each cluster, the clients collaborate to train a group model. Unlike clustering, FedFomo [17] probabilistically selects batches of uploaded models and distributes them to clients, which each calculate model weights and perform weighted aggregation to obtain tailored models. However, sending multiple models to the clients in each communication round introduces high communication overhead. Compared to FedFomo, FedCAP performs customized aggregation on the server side without incurring any additional communication overhead.

Byzantine-robust Federated Learning. Given the distributed nature of FL, it is vulnerable to Byzantine threats [13], [34], [35]. To defend against Byzantine attacks, a series of server-side AGRs built on top of averaged aggregation have been proposed (e.g., Krum [21], Multi-Krum [21], Median [20], RFA [36], Trimmed Mean [20], etc.). Since these AGRs assume that all clients' data are IID, their robustness is less effective in non-IID settings. For non-IID defenses, in FLTrust [18], the server filters or processes abnormal model updates by checking the magnitude and direction of the client-uploaded model updates. However, FLTrust assumes the server holds a clean dataset to boost trust, which violates FL's privacy principles. To mitigate gradient heterogeneity, Karimireddy et al. [24] suggested dividing the uploaded model updates into several buckets before aggregation, averaging s model updates within each bucket, and then using AGRs to aggregate the updates across buckets. To address the issue of the curse of dimensionality [37], which enables malicious gradients to circumvent defenses that aggregate all honest gradients, GAS [25] splits high-dimensional gradients into p low-dimensional sub-vectors, scores them with a robust AGR, and aggregates the gradients identified as honest based on low gradient scores.

For client-side defenses, by training a personalized model for each client with a regularizer controlling the distance between the personalized model and the global model, Ditto [15] mitigates the impact of data heterogeneity and attacks to some extent. However, Ditto cannot generalize well across various non-IID settings and different types of attacks due to the difficulty in balancing client personalization and learning from global knowledge.

A naive combination of server-side AGRs and client personalization (e.g., Ditto) cannot fully mitigate the impact of data heterogeneity and attacks in non-IID settings. Compared to the aforementioned methods, FedCAP is proposed for ensuring unified robustness against both data heterogeneity and attacks in non-IID settings.

III. Background and Motivation

A. *Federated Learning*

The original goal of FL is to maintain user data locally while coordinating clients to train a single global model. The vanilla FL algorithm (i.e., FedAvg [9]) consists of three steps: In each round t, the server distributes the global model $\boldsymbol{w}^t$ to participating clients, which then perform local training with their private data, uploading their updated models to the server. The server performs weighted averaging aggregation to get the updated global model $\boldsymbol{w}^{t+1}$. The optimization problem of FedAvg can be expressed as follows:

$$\boldsymbol{w}_k^t = \arg\min_{\boldsymbol{w}} \mathcal{L}_k(\boldsymbol{w}) \ \ \text{(initialized with } \boldsymbol{w}^t\text{)}, \tag{1}$$

where $\mathcal{L}_k = \frac{1}{|\mathcal{D}_k|}\sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$, $\boldsymbol{w}^{t+1} \leftarrow \sum_{k=1}^{N} p_k \boldsymbol{w}_k^t$. Here, $\mathcal{S}_{(k\in S)}$ represents the set of clients, N is the number of participants in each round, $\mathcal{D}_k$ denotes the training dataset of client k, ℓ is the loss function, $(\boldsymbol{x}_i, y_i)$ denotes a sample pair, and $p_k = \frac{|\mathcal{D}_k|}{|\mathcal{D}|}$ represents the aggregation weight assigned to client k. However, real-world user data often exhibits non-IID characteristics with various distribution skews [10], making it challenging for a single global model to effectively generalize across heterogeneous data.

Personalized Federated Learning. To tackle the challenge of data heterogeneity, several personalized FL algorithms have been proposed. The optimization objective of the personalized model can be formulated in a general form as:

$$\boldsymbol{v}^{t+1} = \arg\min_{\boldsymbol{v}} \mathcal{L}(\boldsymbol{v}) + \lambda\mathcal{R}(\boldsymbol{v}, \boldsymbol{w}^*), \tag{2}$$

where $\boldsymbol{v}$ is initialized with $\boldsymbol{v}^t$ and represents the personalized model, $\boldsymbol{w}^*$ denotes the global knowledge, such as the global model $\boldsymbol{w}^t$ [15], $\mathcal{R}$ denotes the regularizer, and λ denotes the regularization factor controlling the extent to which the personalized model $\boldsymbol{v}$ references the global knowledge $\boldsymbol{w}^*$.

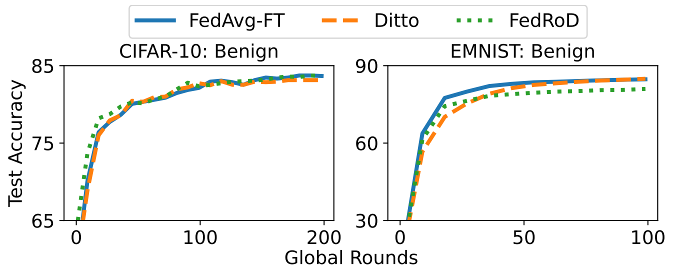


Fig. 1. Model performance comparison of SOTA FL methods

B. *Limitations of the SOTA*

1) Limitations of Existing Personalized FL Methods: To explore the limitations of representative FL approaches discussed in the related work, we evaluate the performance of FedAvg-FT [32], Ditto [15], and FedRoD [16] in benign scenarios using both CIFAR-10 [38] and EMNIST [39]. For CIFAR-10, we adopt the pathological non-IID setting [9] to divide the data into 20 clients with a participating ratio of 1.0, where each client's data contains only two class types. For EMNIST, following a previous study [30], we distribute the data across 100 clients with a participation ratio of 0.2, allocating 20% of the data as IID to each client and sorting the remaining 80% based on labels.[1] To simulate a realistic setting, the size of each client's sample is limited to a few hundred. We report the average test accuracy of personalized models. From Fig. 1, we summarize as follows:

- In benign scenarios, the performance of the above personalized FL methods is comparable to or even worse than FedAvg-FT. This suggests that they are effective only in specific non-IID settings and struggle to adapt well to various non-IID settings with different distribution skew. Similar statements can be found in [16], [32].

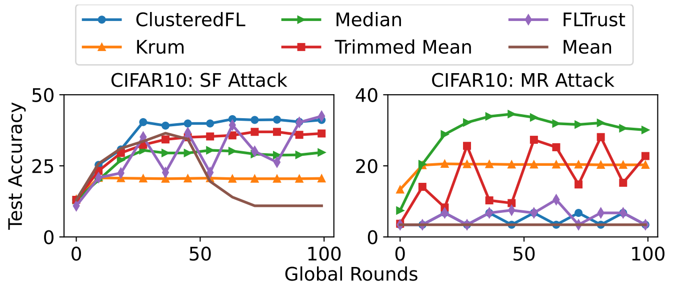


Fig. 2. Model performance comparison of robust FL methods

2) Limitations of Existing Robust FL Methods: To assess the robustness of existing robust FL methods, we evaluate the test accuracy of the global model on CIFAR-10 [38]. Specifically, we examine the performance of FedAvg [9], FLTrust [18], and AGRs including Krum [21], Median [20], Trimmed Mean [20], and ClusteredFL [22] under Sign Flipping (SF) and Model Replacement (MR) attacks [40] detailed in Section VI-A4. Fig. 2 shows that:

[1]If not specifically mentioned, these parameter settings are used by default for all experiments in Section III.

- Existing robust FL methods exhibit varying degrees of defense against attacks but often sacrifice valuable knowledge, as they struggle to distinguish malicious clients from benign ones in non-IID settings. Similar conclusions have been made in [24], [25].

C. *Insights and Motivations*

1) Good Becomes Better; Bad Becomes Worse: Despite the statistical heterogeneity of data among clients, inherent similarities (e.g., common features) in data distributions still exist among some clients [41]. Hence, promoting collaboration among similar clients can be advantageous. As the number of global rounds increases, models among similar clients become more similar and are less influenced by data heterogeneity. Moreover, by identifying anomalies in malicious behaviors, we can inhibit the cooperation between benign and malicious clients, safeguarding benign models from attacks. With more global rounds, malicious models are updated in a worse direction, accelerating their deterioration.

2) Abnormal Euclidean Norm of Malicious Model Updates: To investigate the impact of Byzantine attacks on FL training, we evaluate FedAvg [9] under SF [40] and MR [15] attacks using CIFAR-10 [38]. Fig. 3 illustrates that in both attack scenarios, the Euclidean norm of model updates uploaded by malicious clients increases dramatically as the number of global rounds increases, deepening the impact of attacks on the global model. Ultimately, FL training becomes dominated by attacks, resulting in a substantial decrease in the model performance of benign clients.

These insights motivate us to propose the customized model aggregation rule, design the model update calibration mechanism, and develop the anomaly detection module. These innovations facilitate collaboration among similar clients, accurately capture differences in model updates between benign and malicious clients, and empower the server to identify and remove malicious clients.

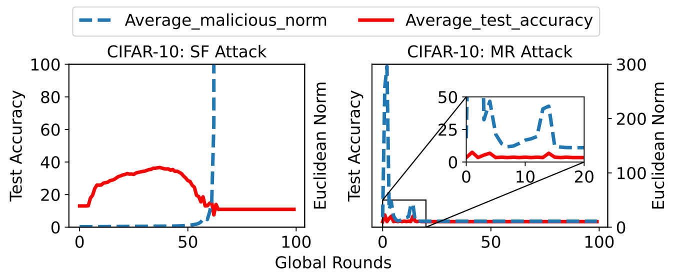


Fig. 3. Impact of model poisoning attacks on Euclidean norm of updates

IV. Problem Formulation

A. *Aggregation Function*

In Eq. 1, the aggregation rule ignores the contributions of clients to each other, leading to the global model that unfairly favors clients with more samples. Our approach falls into the category of multi-model FL, where the server aggregates multiple models. Unlike Eq. 1, FedCAP customizes the aggregation weights for each client. The aggregation of client k's customized model $\hat{\boldsymbol{w}}_k$ can be expressed as:

$$\hat{\boldsymbol{w}}_k \leftarrow \sum_{i=1}^{N} p_{ki} \boldsymbol{w}_i, \tag{3}$$

where p_{ki} denotes the contribution between client k and i. Eq. 3 takes into account inter-client contributions, enabling the customized model of client k to better match its data distribution. Further details are discussed in Section V-A.

B. *Threat Model*

In this work, our focus is on enhancing Byzantine-robustness in FL against poisoning attacks.

Adversary's Goal. The objective of the attack is to disrupt the FL training process, resulting in a significant degradation in model performance on any test inputs (i.e., untargeted poisoning attack [13]). In real-world scenarios, such attacks could cause FL systems to crash, leading to inaccurate model inference in downstream tasks (e.g., disease diagnosis), which could result in immeasurable losses [42].

Adversary's Capabilities. Adversaries may intrude into FL systems by injecting fake clients or compromising benign ones. Considering the attack's cost and feasibility in real-world scenarios, the proportion of malicious clients typically does not exceed 50% [13]. Given their known knowledge, adversaries can manipulate the FL training process by uploading arbitrary or finely crafted malicious model updates to the server, affecting the model aggregation.

Adversary's Knowledge. We consider attack scenarios where adversaries have partial knowledge, including model updates, local data, and local update rules from malicious clients. Despite its limited practical applicability, we further introduce a full-knowledge scenario to explore the upper bound of Byzantine robustness in our method. Under this assumption, adversaries have full knowledge of all clients, including benign ones, enabling them to design stronger and adaptive attacks.

V. Design of FedCAP

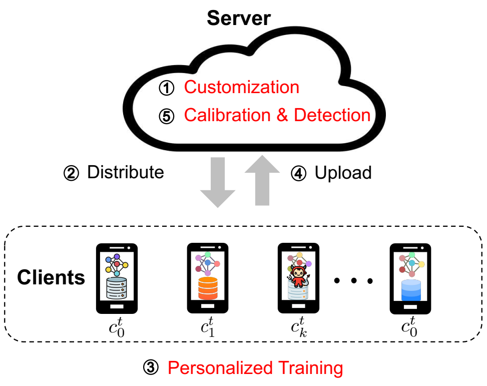
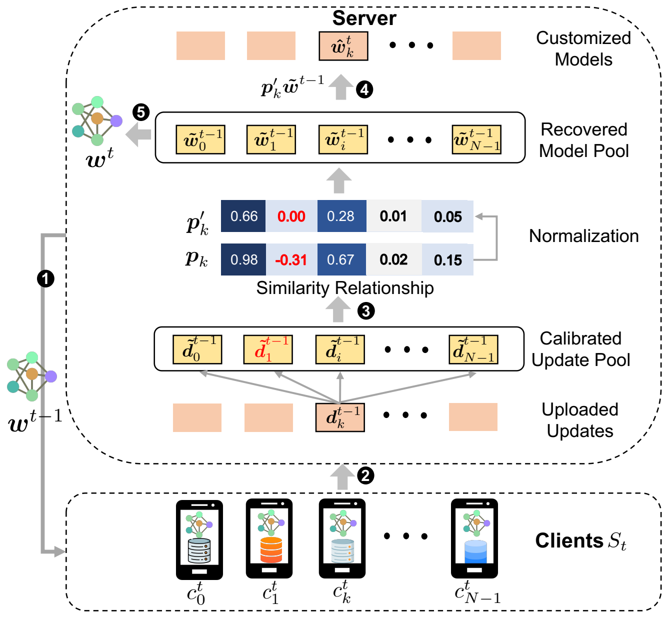


Fig. 4. Workflow of FedCAP

Fig. 4 presents the workflow of FedCAP, comprising three main modules highlighted in red: model customization, personalized training, and model update calibration and detection. In each round, the server begins by customizing the models based on historical knowledge, which includes a recovered model pool and a calibrated update pool, along with the model updates uploaded by the clients (V-A). Subsequently, these customized models are distributed to the participating clients. Upon receiving their respective customized models, the clients perform local updating and personalized training using their private data (V-B). The server then calibrates the uploaded model updates to capture differences between clients, checks the calibrated model updates to detect malicious behavior, and updates historical knowledge (V-C). This iterative process continues for a certain number of rounds until the target accuracy is achieved or a pre-set number of rounds is reached.

A. Model Customization



Server
$\hat{w}_k^t$
Customized Models
$p'_k \tilde{w}^{t-1}$ ❹
❺
w^t
$\tilde{w}_0^{t-1}$ $\tilde{w}_1^{t-1}$ $\tilde{w}_i^{t-1}$ $\tilde{w}_{N-1}^{t-1}$
Recovered Model Pool
p'_k 0.66 0.00 0.28 0.01 0.05
p_k 0.98 -0.31 0.67 0.02 0.15
Normalization
Similarity Relationship
❸
❶
$\tilde{d}_0^{t-1}$ $\tilde{d}_1^{t-1}$ $\tilde{d}_i^{t-1}$ $\tilde{d}_{N-1}^{t-1}$
Calibrated Update Pool
d_k^{t-1}
Uploaded Updates
w^{t-1}
❷
Clients S_t
c_0^t c_1^t c_k^t c_{N-1}^t

Fig. 5. Model customization

We assume that FL training has progressed to round $t (t > 0)$ and client k is joining FL training, where $k \in S_t$ and $|S_t| = N$. The model customization for client k includes the following steps:

Collection. The server collects the model update $\boldsymbol{d}_k^{t-1}$ from client k (as shown in ❶-❷ in Fig. 5). Depending on whether client k participated in the previous round or is a new client, the server handles it differently:

- If client k participated in round $t-1$ (i.e., $k \in S_{t-1}$), the server locally keeps the calibrated update pool $\{\tilde{\boldsymbol{d}}_i^{t-1}\}_{i \in S_{t-1}}$. In this case, there is no need to collect the model update $\boldsymbol{d}_k^{t-1}$ from client k for round t (i.e., $\boldsymbol{d}_k^{t-1} = \tilde{\boldsymbol{d}}_k^{t-1}$), which helps save communication overhead. Since we cannot directly calculate the contribution of client k to itself, we define the contribution of client k to itself (i.e., $p'_{k,k}$ in Eq. 5) as the weight factor ϕ, and the remaining $1-\phi$ is assigned based on Eq. 4 and Eq. 5. The impact of ϕ's value on model performance will be analyzed in Section VI-D2.
- If client k did not join in round $t-1$ or is a new client, the server sends the global model $\boldsymbol{w}^{t-1}$ to client k, and the client returns the model update $\boldsymbol{d}_k^{t-1}$ to the server, where $\boldsymbol{d}_k^{t-1} = \boldsymbol{w}_k^{t-1} - \boldsymbol{w}^{t-1}$.

Customized Aggregation. Then, the server performs the customized model aggregation (as shown in ❸-❹ in Fig. 5). It first computes the cosine similarity (Eq. 4) between $\boldsymbol{d}_k^{t-1}$ and the calibrated update pool $\{\tilde{\boldsymbol{d}}_i^{t-1}\}_{i \in S_{t-1}}$:

$$p_{k,i(k \neq i)} = \frac{< \boldsymbol{d}_k^{t-1}, \tilde{\boldsymbol{d}}_i^{t-1} >}{||\boldsymbol{d}_k^{t-1}|| \cdot ||\tilde{\boldsymbol{d}}_i^{t-1}||}, \tag{4}$$

where the contribution $p_{k,i}$ between client k and i is determined by the similarity between calibrated model updates. The reason is that the similarity between model updates reflects the similarity of user data distribution [22], [33]. Clients with higher similarity contribute more valuable knowledge to each other. Therefore, when customizing aggregation weights, larger weights will be assigned to similar clients. In this way, the impact of data heterogeneity on customized models is mitigated by customized aggregation.

Since the cosine similarity takes values in the range of $[-1, 1]$, the softmax function is introduced for normalization (Eq. 5) to ensure that the sum of aggregation weights is 1, and the weight of each client is non-negative.

$$p'_{k,i(k \neq i)} = \frac{e^{\alpha p_{k,i}}}{\sum_i^N e^{\alpha p_{k,i}}}. \tag{5}$$

The scale factor α in Eq. 5 controls the sensitivity of the weight vector $\boldsymbol{p}'_k$ to the similarity vector $\boldsymbol{p}_k$. When $\alpha = 0$, $p'_{k,i} = 1/N$, and all participants have the same weight. Thus, by controlling the value of scaling factor α, our customized aggregation can adapt to various degrees of data heterogeneity. Moreover, in attack scenarios, due to the low similarity between the model updates of malicious and benign clients, a large α is suggested to amplify the penalty for malicious clients, ensuring that the corresponding weights of the malicious clients after normalization converge to 0. This prevents the aggregation process of the customized models of benign clients from being interfered with by malicious clients.

Once the aggregation weight vector $\boldsymbol{p}'_k$ is obtained, the server aggregates the recovered model pool $\{\tilde{\boldsymbol{w}}_i^{t-1}\}_{i \in S_{t-1}}$ based on $\boldsymbol{p}'_k$ to customize the model $\hat{\boldsymbol{w}}_k^t$ for client k (Eq. 3).

Global Model Updating. At the same time, the server aggregates the recovered model pool $\{\tilde{\boldsymbol{w}}_i^{t-1}\}_{i \in S_{t-1}}$ to update the global model (as shown in ❺ in Fig. 5). Even though FedCAP provides the customized model for each client, it still aggregates the global model in each round for subsequent model update calibration (see Section V-C). The aggregation of the global model is formulated as follows:

$$\boldsymbol{w}^t \leftarrow \sum_{i}^{S_{t-1}} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} \tilde{\boldsymbol{w}}_i^{t-1}. \tag{6}$$

B. Personalized Training

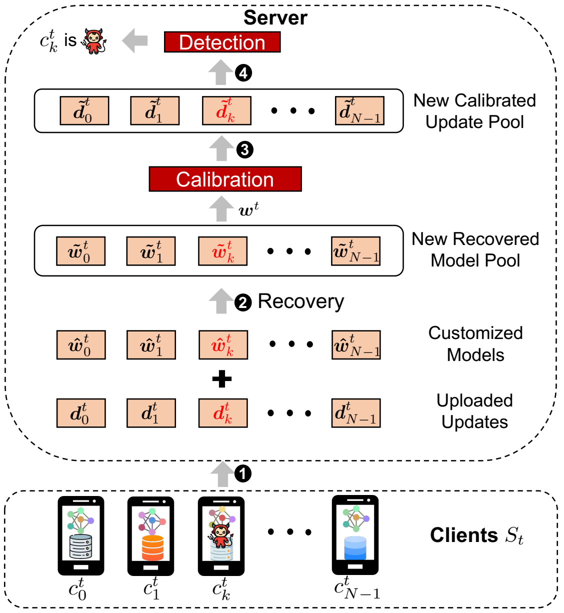
Unlike the global model aggregated in single-model FL, the customized model $\hat{\boldsymbol{w}}_k^t$ better matches the data distribution of client k. However, considering that the customized model may still be affected by slight data heterogeneity or attacks in situations where no explicit similarity relationship exists among clients, we additionally train a personalized model for each client to further mitigate the impact of heterogeneity or attacks. The optimization objective for personalized model of client k is formulated as follows:

$$\boldsymbol{v}_k^{t+1} = \arg\min_{\boldsymbol{v}} \mathcal{L}_k(\boldsymbol{v}) + \frac{\lambda}{2}\|\boldsymbol{v} - \hat{\boldsymbol{w}}_k^t\|_2^2, \tag{7}$$

where $\boldsymbol{v}$ is initialized with $\boldsymbol{v}_k^t$. Eq. 7 follows the general form of objective functions for personalized FL (Eq. 2). Distinguishing from the global model $\boldsymbol{w}^t$, here, $\boldsymbol{w}^*$ in Eq. 2 denotes the customized model $\hat{\boldsymbol{w}}_k^t$, and the regularizer $\mathcal{R}$ uses the square of the L2-norm. The regularization factor λ controls the extent to which client k's personalized model $\boldsymbol{v}$ references the customized model $\hat{\boldsymbol{w}}_k^t$, further mitigating the impact of data heterogeneity or attacks.

In our implementation, client k iteratively updates the personalized model $\boldsymbol{v}_k^t$ and the customized model $\hat{\boldsymbol{w}}_k^t$. When updating $\boldsymbol{v}_k^t$ with Eq. 7, the parameters of $\hat{\boldsymbol{w}}_k^t$ are frozen. Then, $\hat{\boldsymbol{w}}_k^t$ is updated in the same mini-batch SGD: $\boldsymbol{w}_k^t = \arg\min_{\boldsymbol{w}} \mathcal{L}_k(\boldsymbol{w})$, where $\boldsymbol{w}$ is initialized with $\hat{\boldsymbol{w}}_k^t$.

C. Model Update Calibration and Detection



Server
c_k^t is
Detection
❹
$\tilde{\boldsymbol{d}}_0^t$ $\tilde{\boldsymbol{d}}_1^t$ $\tilde{\boldsymbol{d}}_k^t$ $\tilde{\boldsymbol{d}}_{N-1}^t$
New Calibrated Update Pool
❸
Calibration
$\boldsymbol{w}^t$
$\tilde{\boldsymbol{w}}_0^t$ $\tilde{\boldsymbol{w}}_1^t$ $\tilde{\boldsymbol{w}}_k^t$ $\tilde{\boldsymbol{w}}_{N-1}^t$
New Recovered Model Pool
❷ Recovery
$\hat{\boldsymbol{w}}_0^t$ $\hat{\boldsymbol{w}}_1^t$ $\hat{\boldsymbol{w}}_k^t$ $\hat{\boldsymbol{w}}_{N-1}^t$
Customized Models
$\boldsymbol{d}_0^t$ $\boldsymbol{d}_1^t$ $\boldsymbol{d}_k^t$ $\boldsymbol{d}_{N-1}^t$
Uploaded Updates
❶
Clients S_t
c_0^t c_1^t c_k^t c_{N-1}^t

Fig. 6. Model update calibration and detection

After client k finishes updating the customized model, it returns the model update $\boldsymbol{d}_k^t$ to the server, where $\boldsymbol{d}_k^t = \boldsymbol{w}_k^t - \hat{\boldsymbol{w}}_k^t$ (as shown in ❶ in Fig. 6). In FedCAP, since the server customizes a unique model for each participating client, the starting points of local model updating in each round differ among clients[2], making it challenging for the server to directly use the uploaded model updates $\{\boldsymbol{d}_k^t\}_{k\in S_t}$ to measure similarity relationships among clients during customized aggregation. To eliminate this inconsistency, we propose the model update calibration mechanism (as shown in ❸ in Fig. 6) that leverages the global model aggregated in each round (see Eq. 6 in Section V-A for details) as a common reference point to calibrate the model updates uploaded by the clients.

Recovery. Before calibration, the server needs to recover the locally updated model $\tilde{\boldsymbol{w}}_k^t$ of client k based on the customized model $\hat{\boldsymbol{w}}_k^t$ and the uploaded model update $\boldsymbol{d}_k^t$ (as shown in ❷ in Fig. 6), which can be formulated as $\tilde{\boldsymbol{w}}_k^t = \hat{\boldsymbol{w}}_k^t + \boldsymbol{d}_k^t$.

Calibration. The calibration process of the model update $\boldsymbol{d}_k^t$ uploaded by client k can be represented as $\tilde{\boldsymbol{d}}_k^t = \tilde{\boldsymbol{w}}_k^t - \boldsymbol{w}^t$. After calibration, the server can not only accurately capture the similarity relationships among benign clients, but it can also differentiate malicious model updates from benign ones in non-IID settings. Here are some insights into the functionality of calibration in attack scenarios.

Specifically, we find that cosine similarity can only measure the directional differences of the uploaded model updates, which is why FLTrust [18] and ClusteredFL [22] struggle to resist attacks targeting the magnitude of model updates (e.g., MR attack) in non-IID settings, as shown in Section III-B2. However, even though FedCAP also employs the same distance measurement criterion, it can effectively resist attacks targeting either the magnitude or direction of model updates. For example, when an adversary launches the model poisoning attack targeting the magnitude of the model update (e.g., MR attack), after calibration, the direction of the uploaded model update will be changed. This enables the distance measurement (i.e., cosine similarity) to capture the difference in calibrated model updates between benign and malicious clients.

Detection. For detection, the server calculates the Euclidean norm of the calibrated model update $\tilde{\boldsymbol{d}}_k^t$ for client k (as shown in ❹ in Fig. 6). If the predetermined detection threshold T_{norm} is exceeded, the server will recognize client k as a malicious client and remove it permanently.

D. Algorithm of FedCAP

Alg. 1 outlines the entire training process of FedCAP. When $t = 0$, the server initializes the customized models $\{\hat{\boldsymbol{w}}_k^0\}_{k\in S_0}$ with $\boldsymbol{w}^0$ and distributes them to the corresponding participants S_0 (Line 9-12). Clients perform local updating to obtain updated local models $\{\boldsymbol{w}_k^t\}_{k\in S_t}$ and personalized models $\{\boldsymbol{v}_k^{t+1}\}_{k\in S_t}$ (Line 14). Subsequently, clients upload their model updates $\{\boldsymbol{d}_k^t\}_{k\in S_t}$ to the server (Line 15). The server recovers the updated models of clients, calibrates uploaded

[2]In FedAvg, the starting points of local model updating in each round are the same global model for all clients.

Algorithm 1: FedCAP

```
1  Input: communication rounds $T$, client set $S$, initial global model $\boldsymbol{w}^0$ and personalized models $\{\boldsymbol{v}_k^0\}_{k\in S}$.
2  Output: model pool $\{\tilde{\boldsymbol{w}}_k^{T-1}\}_{k\in S_{T-1}}$, calibrated model update pool $\{\tilde{\boldsymbol{d}}_k^{T-1}\}_{k\in S_{T-1}}$, global model $\boldsymbol{w}^{T-1}$ and personalized models $\{\boldsymbol{v}_k^T\}_{k\in S}$.
3  for $t = 0, \ldots, T-1$ do
4      /* Server randomly selects a subset of clients $S_t$.*/
5      if $t \neq 0$ then
6          $\{\hat{\boldsymbol{w}}_k^t\} \leftarrow$ Customize($\boldsymbol{w}^{t-1}, \{\tilde{\boldsymbol{d}}_i^{t-1}\}, \{\tilde{\boldsymbol{w}}_i^{t-1}\}$)
7          $\boldsymbol{w}^t \leftarrow$ GlobalModelUpdating($\{\tilde{\boldsymbol{w}}_i^{t-1}\}$)
8      else
9          /* Server initializes customized models*/
10         $\{\hat{\boldsymbol{w}}_k^0\}_{(k\in S_0)}$, where $\hat{\boldsymbol{w}}_k^0 \leftarrow \boldsymbol{w}^0$.
11     end
12     /* Server distributes $\{\hat{\boldsymbol{w}}_k^t\}_{(k\in S_t)}$ to clients $S_t$.*/
13     for each client $k \in S_t$ do
14         $\boldsymbol{w}_k^t, \boldsymbol{v}_k^{t+1} \leftarrow$ ClientUpdate($\hat{\boldsymbol{w}}_k^t, \boldsymbol{v}_k^t$)
15         $\boldsymbol{d}_k^t \leftarrow$Poison($\boldsymbol{w}_k^t - \hat{\boldsymbol{w}}_k^t$)
16     end
17     /* Server calibrates uploaded model updates and detects whether client k is malicious.*/
18     $\{\tilde{\boldsymbol{w}}_k^t\} \leftarrow$ Recover($\{\hat{\boldsymbol{w}}_k^t\}, \{\boldsymbol{d}_k^t\}$)
19     $\{\tilde{\boldsymbol{d}}_k^t\} \leftarrow$ Detect(Calibrate($\{\tilde{\boldsymbol{w}}_k^t\}, \boldsymbol{w}^t$))
20 end
```

model updates, calculates the Euclidean norm of the calibrated model updates to detect whether the clients are malicious or benign, and updates the recovered model pool $\{\tilde{\boldsymbol{w}}_k^t\}_{k\in S_t}$ and the calibrated update pool $\{\tilde{\boldsymbol{d}}_k^t\}_{k\in S_t}$ (Line 18-19).

It's noteworthy that when $t = 0$, all participants have identical parameters for the customized models $\{\hat{\boldsymbol{w}}_k^0\}_{k\in S_0}$. Therefore, the parameters of the calibrated model update of client k are identical to those of its uploaded model update (i.e., $\tilde{\boldsymbol{d}}_k^t = \boldsymbol{d}_k^t$). When $t > 0$, the server customizes models for clients through Section V-A. The subsequent steps remain consistent with those at $t = 0$. Upon FL training, the server holds the recovered model pool $\{\tilde{\boldsymbol{w}}_k^{T-1}\}_{k\in S_{T-1}}$, the calibrated update pool $\{\tilde{\boldsymbol{d}}_k^{T-1}\}_{k\in S_{T-1}}$, and the global model $\boldsymbol{w}^{T-1}$.

VI. Experiment Evaluations

A. Experiment Setups

1) Datasets: We use two image classification datasets, CIFAR-10 [38] and EMNIST [39], as well as a human activity recognition dataset, WISDM [41]. The CIFAR-10 dataset contains images from 10 classes. We adopt a pathological non-IID setting [9] that introduces label distribution skew. Specifically, we define 20 clients, each with a balanced number of samples but imbalanced classes (2 classes per client). The EMNIST dataset comprises images of 62 different digits and letters. Following previous studies [30], [43], we employ a non-IID setting to divide the data (commonly seen in cross-silo settings). Specifically, we define 100 clients and allocate data based on digits, lowercase letters, and uppercase letters, forming 3 distinct groups. Within each group, client data consists of 80% samples from dominant classes and 20% samples from all classes, leading to imbalanced sample distributions among the groups. The WISDM data is collected from 36 user devices and 6 activity classes. We employ the default data distribution setting as the user-specific physiological and environmental variations introduce statistical heterogeneity (i.e., feature distribution skew) in the collected activity data.

For all datasets, we split the client data into training and test sets with a ratio of 0.75. To simulate scenarios where user sample sizes are limited, we restrict the number of samples for all clients to be on the order of hundreds.

2) Models: For all datasets, the models consist of two convolutional layers (with filter numbers ranging from 32 to 64 for CIFAR-10 and WISDM, and from 16 to 32 for EMNIST), followed by a fully connected layer (with 64 units for CIFAR-10 and WISDM, and 128 units for EMNIST), and an output layer. Note that considering the limited resources of user devices in real-world scenarios and the primary focus of this work on designing an FL algorithm, we do not explore other potentially better-performing models.

3) Attack Methods: Seven attacks are described below.

Label Flipping (LF) [40]: Malicious clients train models on manipulated data. The original data labels are flipped by $y_i' \leftarrow (y_i + 1)\%C$, where C denotes the number of classes.

Sign Flipping (SF) [40]: Malicious clients flip the signs of their model updates before uploading.

Model Replacement (MR) [15]: Malicious clients scale up model updates by N times before uploading. By default, N is the number of participants per round.

A Little is Enough (LIE) [26]: Malicious clients calculate mean and standard deviation for each coordinate over participant updates and set fake updates within the range of $(\mu_i - z^{max}\delta_i, \mu_i + z^{max}\delta_i)$, where z^{max} is obtained from the Cumulative Standard Normal Function.

Min-Max [27]: Min-Max attack optimizes a malicious gradient, ensuring the maximum distance between it and any benign gradient remains within the upper bound set by the largest distance between any two benign gradients.

Min-Sum [27]: Min-Sum attack optimizes a malicious gradient, ensuring the sum of squared distances of the malicious gradient from all the benign gradients remains within the upper bound set by the sum of squared distances of any benign gradient from the other benign gradients. The optimization problems for the Min-Max (Eq. 9) and Min-Sum (Eq. 10) attacks can be formulated as follows:

$$\boldsymbol{d}_m = f_{\text{avg}}\left(\boldsymbol{d}_{\{i\in[N]\}}\right) + \gamma \boldsymbol{d}_p, \tag{8}$$

$$\underset{\gamma}{\text{argmax}} \max_{i\in[N]} \|\boldsymbol{d}_m - \boldsymbol{d}_i\|_2 \leq \max_{i,j\in[N]} \|\boldsymbol{d}_i - \boldsymbol{d}_j\|_2, \tag{9}$$

$$\underset{\gamma}{\text{argmax}} \sum_{i\in[N]} \|\boldsymbol{d}_m - \boldsymbol{d}_i\|_2^2 \leq \max_{i\in[N]} \sum_{j\in[N]} \|\boldsymbol{d}_i - \boldsymbol{d}_j\|_2^2, \tag{10}$$

where $\boldsymbol{d}_m$ represents the malicious update, and $\boldsymbol{d}_p$ represents the perturbation vector. Following a previous study [27], we choose $\boldsymbol{d}_p$ as -$std(\boldsymbol{d}_{\{i\in[N]\}})$.

Inner Product Manipulation (IPM) [28]: IPM attack aims to achieve a negative inner product between the true mean of the updates and the aggregation result, ensuring that the model update in the direction of gradient ascent.

$$\frac{1}{N}\sum_{i\in[N]}\Delta_i = \frac{N-M(1+\epsilon)}{N(N-M)}\sum_{m\in[M]}\Delta_m, \tag{11}$$

where N denotes the number of participants, and M denotes the number of malicious clients. To ensure that $\frac{N-M(1+\epsilon)}{N(N-M)} < 0$, by default, we set ϵ to N [44].

4) Baselines: To evaluate the effectiveness of personalized FL methods in various non-IID settings, we compare the performance of FedCAP with the six baselines: **Local Training**, **FedAvg** [9], **FedAvg with Fine-tuning** (FedAvg-FT [32]), **Ditto** [15], **FedRoD** [16], and **FedFomo** [17] discussed in the related work.

To further validate the robustness of FedCAP, we compare it with **FLTrust** [18] and the following five AGRs. In **Multi-Krum (M-Krum)** [21], the server averages the top Q models with the smallest scores to obtain the updated global model, with this work using $Q = N - M$. **Median** [20] takes the coordinate-wise median of the participants' client model parameters. **RFA** [36] takes the geometric-wise median of the participants' client model parameters. In **Trimmed Mean (Trim.)** [20], the server sorts the model parameters of the participating clients by coordinates, then averages the remaining parameters after removing the Q largest and Q smallest values, where $Q = \lfloor \frac{M}{2} \rfloor$. Using **ClusteredFL** [22], the server identifies the optimal partitioning of clients into two clusters, and the cluster with the fewest clients is considered malicious. The server then aggregates the models within the remaining cluster. Considering some AGRs are IID defenses, we further combine them with two SOTA non-IID defenses (i.e., **GAS** [25] and **Bucketing** [24]) to explore their robustness in non-IID settings.

5) Parameter Settings: Unless specified otherwise, in all experiments, we fix the batch size to 10, the learning rate to 0.01, global rounds to 100, and the epoch to 5. For CIFAR-10 and WISDM, the proportion of participating clients per round is set to 1.0, while for EMNIST, it is set to 0.2. In the attack scenarios, we assume a default proportion of malicious clients p_{atk} of 0.3. In addition, we conduct a grid search for hyperparameters for all baselines. More specifically, for Ditto, the search range for λ is $\{0.01, 0.1, 1, 2\}$. For FedFomo, we set the number of models sent to each client to half the proportion of participating clients [17]. For Bucketing, we set s to 2 as per the original paper. For GAS, we search for the optimal number of sub-vectors p in $\{1000, 10000, 100000\}$. As for FedCAP, the search range for λ is $\{0.1, 0.5, 1\}$, for ϕ it is $\{0.1, 0.2, 0.3\}$, and for α, it is $\{2, 5, 10\}$. In all attack scenarios, we fix the value of T_{norm} to 10. This is because as the global rounds increase, the Euclidean norm of the calibrated model updates from malicious clients gradually approaches infinity, so the anomaly detection mechanism is not sensitive to T_{norm}.

6) Evaluation Metric: In all experiments, we report the average **Test Accuracy (TAcc)** of all benign client models in the final global round. If not specifically mentioned, we choose the model with higher average test accuracy between customized models and personalized models.

B. Performance Comparisons

1) Comparing with SOTA FL Baselines: We use the CIFAR-10, EMNIST, and WISDM datasets to compare the performance of FedCAP with other SOTA baselines in the benign scenario, and we also explore their potential defense ability in attack scenarios (i.e., LF, SF, and MR). In Fig. 7, FedCAP outperforms all other baselines in all cases in terms of model accuracy. It exhibits an average accuracy gain of 2% to 23% over other baselines in the benign scenario and an average accuracy gain of 3% to 50% in attack scenarios.

Specifically, in benign scenarios, although Ditto and FedRoD achieve comparable model performance on WISDM to FedCAP, their performance on CIFAR-10 and EMNIST shows a relatively marginal gap compared to FedCAP, averaging about 3%. These results indicate that existing personalized FL algorithms, which rely solely on personalized training, cannot consistently achieve satisfactory performance under varying non-IID settings. In contrast, FedCAP adapts to different levels of data heterogeneity through customized aggregation (with the scale factor α—10 for CIFAR-10 and EMNIST, and 2 for WISDM) and further mitigates the impact of data heterogeneity by incorporating the personalized training mechanism.

In attack scenarios, most methods fail to defend against attacks across heterogeneous datasets. Although FedFomo shows some robustness against attacks, balancing model utility and robustness through client-side personalized aggregation is challenging given the unknown number of malicious clients in real-world scenarios. Compared to others, FedCAP demonstrates strong robustness against all attacks. Especially under the MR and SF attacks, its model performance is on average about 3% to 62% higher than others.

In summary, through the combined effects of customized aggregation and personalized training, FedCAP not only mitigates the impact of data heterogeneity, but it also effectively defends against malicious attacks by incorporating model update calibration and anomaly detection mechanisms. The above results show that FedCAP generalizes well to various data heterogeneity settings and attack scenarios.

2) Comparing with Robust FL Methods: To evaluate the robustness of FedCAP and existing robust FL methods, we use the CIFAR-10, EMNIST, and WISDM datasets to compare their model performances under six attack scenarios. For a fair comparison, we report the average test accuracy of their locally updated models. We do not consider the LF attack as it does not significantly affect most FL baselines in Section VI-B1. As can be seen from Table I, Median resists the attacks to some extent in most cases. ClusteredFL and FLTrust cannot counter the MR attack in non-IID settings because they use cosine similarity as the distance measurement, which cannot distinguish differences in the magnitude of model updates. Trimmed Mean fails to defend against the IPM attack. In contrast, FedCAP outperforms other robust FL methods in all cases with an average of 12% to 27% without a marginal

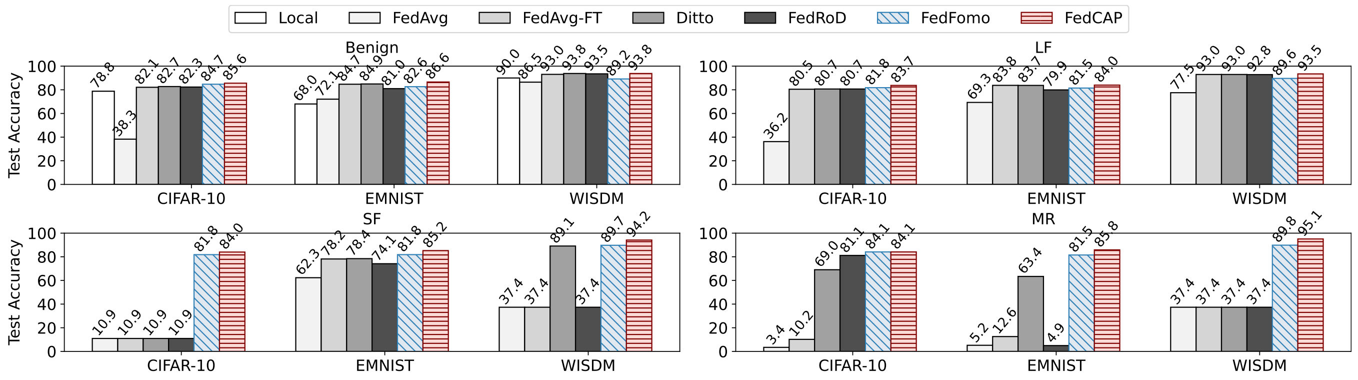


Fig. 7. Model performance comparison of FedCAP with SOTA FL baselines

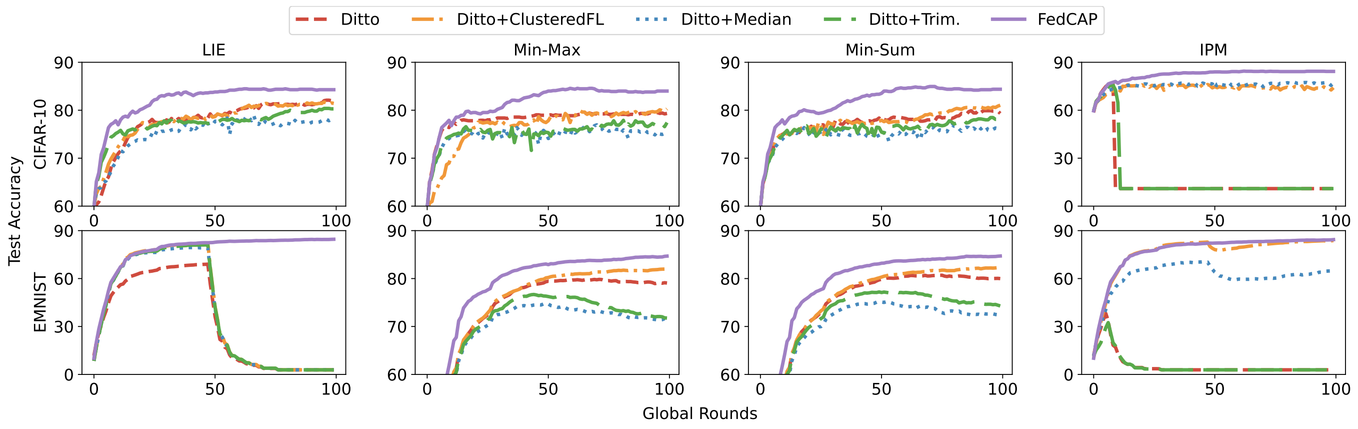


Fig. 8. AGRs augmented with Ditto

accuracy loss compared to benign scenarios. Especially under the LIE attack on EMNIST, all methods except for FedCAP fail to resist the attack. This is attributed to the model update calibration in FedCAP, which helps capture the differences between malicious and benign model updates in non-IID settings, thereby aiding the server in reducing the impact of malicious model updates during customized aggregation.

In sum, the above results demonstrate that FedCAP can achieve superior robustness under non-IID settings and generalize to various attack scenarios.

3) AGRs Augmented with Ditto: Since FedCAP includes the server-side customized aggregation and the client-side personalization, we use the CIFAR-10 and EMNIST datasets to explore whether the combination of AGRs and SOTA personalized FL method Ditto is enough to excel in defending against attacks in non-IID settings. Fig. 8 shows that although the combination of Ditto and ClusteredFL mitigates the impact of attacks such as IPM to some extent, its model performance is still on average about 13% lower than that of FedCAP. Furthermore, in most cases, the model performance of Ditto combined with Median or Trimmed Mean even declines compared to Ditto. The reason is that these AGRs fail to detect malicious clients in non-IID settings, so the global model is still influenced by attacks, resulting in less valuable global knowledge for the personalized model to reference.

TABLE I
MODEL PERFORMANCE COMPARISON OF FEDCAP WITH ROBUST FL METHODS

Dataset	Type	Mean	M-Krum	Median	RFA	Trim.	ClusteredFL	FLTrust	FedCAP
CIFAR-10	SF	10.95	80.57	78.86	78.57	74.38	79.71	82.19	**84.00**
	MR	10.19	82.38	82.38	82.38	81.24	17.24	20.67	**84.10**
	LIE	80.86	79.90	77.24	78.00	80.19	82.57	72.10	**84.95**
	Min-Max	77.24	52.00	69.81	74.76	72.61	78.95	81.05	**84.00**
	Min-Sum	78.48	67.43	66.86	73.71	72.29	78.86	80.19	**84.38**
	IPM	10.95	82.38	75.05	36.48	10.95	42.00	81.14	**84.29**
EMNIST	SF	78.15	81.80	79.55	80.27	79.66	80.04	84.05	**85.24**
	MR	12.56	83.64	84.30	84.97	4.47	2.96	3.01	**85.82**
	LIE	2.87	2.87	2.87	2.87	2.87	2.87	2.87	**85.16**
	Min-Max	76.86	71.08	70.31	71.55	69.59	81.28	83.01	**85.39**
	Min-Sum	78.76	71.68	71.06	71.80	72.02	81.61	83.37	**85.58**
	IPM	2.87	2.06	65.88	19.06	2.87	83.72	84.13	**85.43**
WISDM	SF	37.36	90.23	92.25	91.04	89.53	89.02	93.35	**94.16**
	MR	37.36	93.45	93.66	93.15	91.74	37.36	37.36	**95.07**
	LIE	94.36	94.56	92.75	92.55	93.76	94.36	37.36	**94.56**
	Min-Max	90.13	92.35	91.44	92.65	82.78	91.84	91.54	**94.06**
	Min-Sum	90.23	93.45	91.34	92.75	83.69	93.35	92.15	**94.26**
	IPM	37.36	93.35	81.67	82.38	37.36	37.36	93.25	**94.06**

In contrast, FedCAP not only demonstrates strong robustness in all cases but also converges faster than other methods. This is due to its model update calibration and anomaly detection mechanisms, which enable the server to detect malicious clients and aggregate customized models based on the contributions (i.e., similarities) among benign clients. This allows client personalized models to reference more valuable information from the customized models.

TABLE II
AGRs Augmented with Non-IID Denfenses

Dataset	Method / Attack	MKrum Bucket	MKrum GAS	Median Bucket	Median GAS	RFA Bucket	RFA GAS	FedCA
CIFAR10	LIE	80.57 +0.67	80.86 +0.96	79.43 +2.19	80.00 +2.76	80.38 +2.38	80.19 +2.19	**84.95**
	IPM	10.95 -71.43	82.38 +0	10.95 -64.10	82.76 +7.71	10.95 -25.53	82.76 +46.28	**84.29**
EMNIST	LIE	25.01 +22.14	2.87 +0	27.31 +24.44	2.87 +0	30.28 +27.41	2.87 +0	**85.16**
	IPM	10.60 +8.54	3.96 +1.90	10.60 -55.28	4.02 -61.86	10.60 -8.46	2.87 -16.19	**85.43**
WISDM	LIE	93.57 -0.99	94.71 +0.15	93.19 +0.44	94.14 +1.39	94.26 +1.71	94.86 +2.31	**94.56**
	IPM	38.94 -54.41	91.49 -1.86	38.94 -42.73	92.06 +10.39	37.36 -45.02	93.55 +11.17	**94.06**

4) AGRs Augmented with Non-IID Denfenses: In Section II, we introduced two SOTA non-IID defenses (i.e., Bucketing and GAS). To evaluate the robustness of combining non-IID defenses with AGRs in non-IID settings, we compare the performance of M-Krum, Median, and RFA when combined with Bucketing and GAS using three datasets. Table II presents the model performance and the performance improvement after combination, with negative numbers indicating a performance decline. From Table II, it is evident that under the LIE attack, both non-IID defenses lead to performance improvements, especially on EMNIST. However, under the IPM attack, combining non-IID defenses with AGRs even brings negative effects on model performance, particularly with the Bucketing method. The reason is that before using AGRs for inter-bucket aggregation, average aggregation has occurred in each bucket. This two-step aggregation might cause overly aggressive cancellation, resulting in too many beneficial model updates being excluded from model aggregation, ultimately affecting the model performance. These results indicate that the incorporation of non-IID defenses still cannot generalize across various attacks to assist AGRs in enhancing the robustness.

In contrast, FedCAP withstands attacks in all cases due to its model update calibration mechanism, which enables the server to differentiate malicious model updates from benign ones in non-IID settings. With the enhancement of customized aggregation, FedCAP accelerates the deterioration of malicious client models, ensuring that the anomaly detection mechanism quickly identifies and removes malicious clients, effectively defending against attacks.

C. Robustness of FedCAP

TABLE III
Robustness Analysis of FedCAP

Dataset	Metrics	Benign	SF	MR	LIE	Min-Max	Min-Sum	IPM
CIFAR-10	TAcc	85.60	84.00	84.10	84.95	84.00	84.38	84.29
	DAcc	-	100	100	70	100	100	100
	FPR	-	0	0	0	0	0	0
	FNR	-	0	0	100	0	0	0
EMNIST	TAcc	86.59	85.24	85.82	85.16	85.39	85.58	85.43
	DAcc	-	80	100	81	93	96	98
	FPR	-	0	0	0	0	0	0
	FNR	-	20	0	63.33	23.33	13.33	6.67
WISDM	TAcc	93.78	94.16	95.07	94.56	94.06	94.26	94.06
	DAcc	-	100	100	72.22	100	100	100
	FPR	-	0	0	0	0	0	0
	FNR	-	0	0	100	0	0	0

To further demonstrate the robustness of FedCAP, in addition to evaluating the impact of attacks on model performance (i.e., TAcc), we introduce three robustness metrics to measure the detection abilities of FedCAP. **Detection Accuracy (DAcc)** represents the proportion of clients that are correctly identified as either benign or malicious. **False Positive Rate (FPR)** (or **False Negative Rate (FNR)**) denotes the proportion of benign (or malicious) clients that are incorrectly regarded as malicious (or benign). As can be seen from Table III, in most cases, FedCAP can identify and remove almost all malicious clients without mistakenly identifying benign clients as malicious with the FPR = 0.

In particular, FedCAP fails to identify malicious clients under the LIE attack, with the FNR = 100 on CIFAR-10 and WISDM. The reason is that the collaboration among malicious clients during customized aggregation does not exacerbate the deterioration of malicious models and the Euclidean norm of malicious model updates does not reach the detection threshold T_{norm}, thus not triggering the anomaly detection mechanism. Surprisingly, in this case, the model performance of FedCAP is not significantly affected by the attack, with accuracy loss averaging less than 1% compared to benign scenarios. This is attributed to its ability to isolate malicious model updates from benign ones during customized aggregation, so the aggregated customized models of benign clients are not seriously affected by the attack.

D. Impact of Hyperparameters

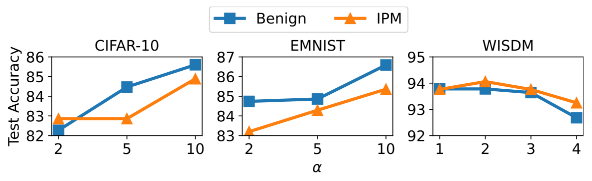


Fig. 9. Impact of scale factor α

1) Impact of Scale Factor α: To analyze the impact of α on the model performance of FedCAP, we conduct experiments using three datasets in both benign and attack scenarios. Here, we only discuss the parameter analysis under the IPM attack, as we find that the conclusions for other attacks are similar to it. As shown in Fig. 9, for the WISDM dataset, as α increases, the model accuracy of FedCAP decreases. This result is attributed to the fact that there are many shared features among user data. For example, all users have highly similar behavior patterns such as walking. Therefore, when customizing aggregation weights, the server should choose a smaller value for α to fairly consider all users and learn more global features. In contrast, for the other two datasets, as α increases, the model accuracy of FedCAP gradually improves. The reason is that there is a higher degree of label distribution skew among clients' data. In this case, choosing a larger value for α allows the server to assign larger weights to clients with similar distributions, preventing interference from other dissimilar or malicious clients, and thus reducing the impact of data heterogeneity or attack.

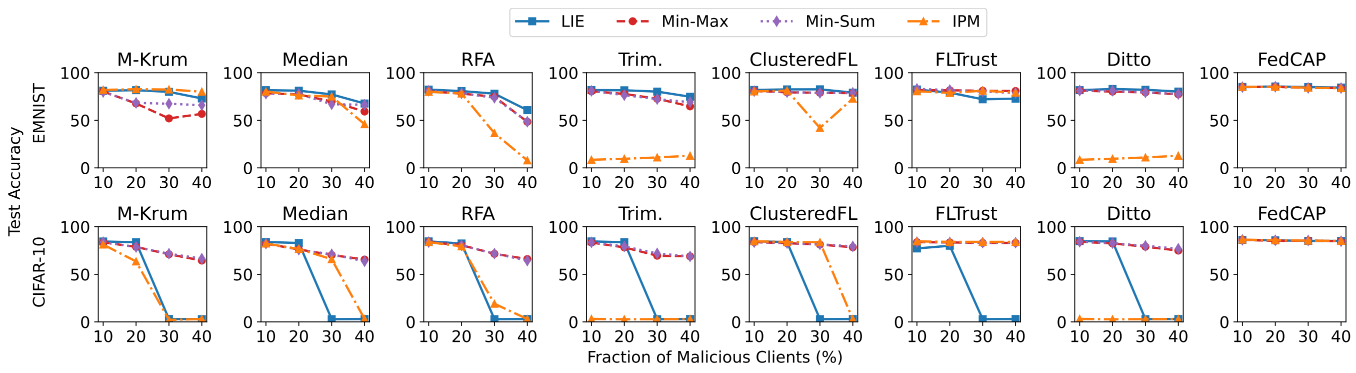


Fig. 10. Impact of the proportion of malicious clients

It is important to note that the choice of α's value depends on the degree of heterogeneity among client data distributions. In real-world attack scenarios, given the unknown number of malicious clients, FedCAP relies on its model update calibration and anomaly detection mechanisms to identify malicious clients, rather than tuning α.

2) Impact of Weight Factor ϕ: To investigate the impact of ϕ on the model performance of FedCAP, we conduct experiments on three datasets in both benign and attack scenarios (e.g., IPM). As shown in Fig. 11, the value of ϕ is inversely proportional to the model accuracy. The reason is that as ϕ increases, the weight corresponding to the client itself becomes larger, which weakens the reference to valuable knowledge from other client models in the model customization, thereby affecting the model performance. Additionally, we find that the performance of FedCAP is relatively robust to the choice of ϕ's value.

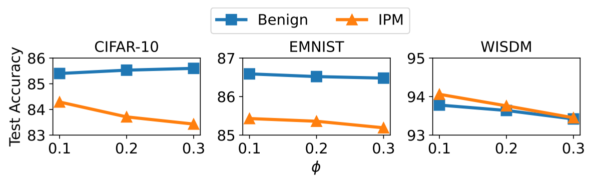


Fig. 11. Impact of weight factor ϕ

3) Impact of Regularization Factor λ: To explore the impact of λ on model performance of FedCAP, we conduct experiments using three datasets in both benign and attack scenarios (e.g., IPM). As shown in Fig. 12, we conclude that appropriately increasing λ not only allows the personalized model training to reference global knowledge from the customized model but also prevents overfitting of the personalized model on the limited data. When λ is too large, the training of the personalized model is influenced by the customized model that may still be affected by data heterogeneity or attack, leading to a decrease in its accuracy.

4) Impact of the Proportion of Malicious Clients: To evaluate the impact of the proportion of malicious clients on the effectiveness of robust FL methods, we use the CIFAR-10 and EMNIST datasets to conduct experiments under four strong attacks (i.e., LIE, Min-Max, Min-Sum, and IPM). Fig. 10 shows that as the proportion of malicious clients increases,

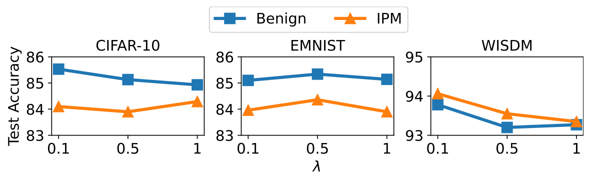


Fig. 12. Impact of regularization factor λ

especially to 30% and 40%, the model performance of most robust FL baselines is significantly affected by attacks and decreases substantially. Moreover, although FLTrust and Ditto resist attacks to some extent on CIFAR-10, their effectiveness significantly is suppressed on EMNIST, indicating that they cannot demonstrate robustness across various attacks in non-IID settings. Conversely, regardless of the proportion of malicious clients, FedCAP exhibits strong robustness and generalizes well across various attacks while maintaining superior model performance.

E. Ablation Study

TABLE IV
ABLATION STUDY ANALYSIS OF FEDCAP COMPONENTS

	Cust. Agg.	Calibration	Pers. Train.	Benign	LIE	Min-Max	Min-Sum	IPM
①	✓			83.53	78.67	48.76	56.29	83.14
②			✓	82.73	82.10	79.33	79.33	10.95
③	✓	✓		85.07	84.29	83.52	83.81	83.71
④	✓		✓	83.60	80.10	69.81	70.95	83.14
⑤(FedCAP)	✓	✓	✓	**85.60**	**84.95**	**84.00**	**84.38**	**84.29**

To verify the necessity of each main component in FedCAP (i.e., customized aggregation, model update calibration, and personalized training) as introduced in Section V, we conduct ablation experiments using CIFAR-10 under four strong attacks and analyze the results in Table IV.

First, ①'s model performance is about 2% lower than ③'s in the benign scenario and about 17% lower on average in attack scenarios. This indicates that without the model update calibration mechanism, the standalone customized aggregation mechanism cannot precisely capture the similarity relationships among benign clients, nor can it differentiate between malicious and benign model updates, leading to a significant drop in the performance of customized models due to the impact of data heterogeneity and attacks.

Second, ②'s model performance is about 3% lower than ⑤'s in the benign scenario, and, while it shows some robustness to attacks other than IPM, the result indicates that the single personalized training mechanism still lacks the robustness to generalize across various attack scenarios.

Third, ③'s model performance is on average less than 1% lower than ⑤'s, suggesting that the combination of customized aggregation and model calibration mechanisms already significantly mitigates the impact of data heterogeneity and attacks, and the addition of the personalized training further enhances model accuracy and robustness.

Last, ④'s model performance is 2% lower than ⑤'s in the benign scenario and about 8% lower on average in attack scenarios. This result shows that simply combining customized aggregation with personalized training mechanisms cannot achieve superior model performance. The incorporation of the model update calibration ensures malicious model updates are differentiated, which enables the server to isolate malicious clients from benign ones during customized aggregation, accelerate the deterioration of malicious models, and trigger the anomaly detection mechanism to identify and remove malicious clients permanently. Based on this, the proposed personalized training module can further improve robustness and model performance, building on the relatively clean customized models.

Therefore, all the proposed modules in FedCAP are essential and contribute to achieving superior accuracy and robustness across various non-IID settings and attacks.

VII. Conclusion and Future Work

In this paper, we proposed FedCAP, a robust FL framework against both data heterogeneity and Byzantine attack. Specifically, we designed a customized model aggregation scheme, which can facilitate collaborative training among similar clients and accelerate the deterioration of malicious models. In addition, we developed a model update calibration mechanism to capture the differences in the direction and magnitude of model updates among clients and an anomaly detection mechanism to help the server quickly identify and permanently remove malicious clients. Extensive experiments demonstrated that FedCAP outperformed the SOTA baselines in terms of model accuracy under several non-IID settings and model robustness under various types of poisoning attacks.

Robustness Analysis. We believe that in strong attack scenarios [26]–[28], even if adversaries know the customized aggregation rule and the model updates of benign clients, it is difficult to design effective adaptive attacks. The reasons are twofold: first, the model updates uploaded by clients are calibrated by the server, making the knowledge adversaries possess about the uploaded model updates outdated. Second, the server aggregates the customized model for each client, making it challenging for adversaries to manipulate malicious model updates that can affect the model customization of all benign clients. In future work, we will conduct more theoretical robustness analysis of FedCAP and explore potential adaptive attack strategies.

Convergence Analysis. In FedCAP, the server's customized aggregation and model update calibration mechanisms dynamically adjust the aggregation process based on incoming updates. This dynamic nature introduces additional layers of complexity for theoretical analysis. In future work, we can decompose the convergence analysis into multiple aspects, such as studying the impact of varying customized aggregation weights on model convergence speed and final performance, theoretically deriving how the model update calibration mechanism affects model convergence, and investigating the impact of regularization factors in the personalized objective function on personalized model training.

Acknowledgment

In this work, Lichao Sun was partially supported by the National Science Foundation (NSF) (CRII-2246067, ATD-2427915, NSF POSE-2346158, and Lehigh Grant FRGS00011497). Xuyu Wang was supported in part by the NSF (CNS-2319343, CNS-2317190, CNS-2415209, and IIS-2306791).

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[2] C. Li, X. Zeng, M. Zhang, and Z. Cao, "Pyramidfl: a fine-grained client selection framework for efficient federated learning," in *ACM MobiCom '22: The 28th Annual International Conference on Mobile Computing and Networking, Sydney, NSW, Australia, October 17 - 21, 2022*. ACM, 2022, pp. 158–171.

[3] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[4] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE journal on selected areas in communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[5] X. Ouyang, Z. Xie, H. Fu, S. Cheng, L. Pan, N. Ling, G. Xing, J. Zhou, and J. Huang, "Harmony: Heterogeneous multi-modal federated learning through disentangled model training," in *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services, MobiSys 2023, Helsinki, Finland, June 18-22, 2023*. ACM, 2023, pp. 530–543.

[6] X. Wang, C. Yang, and S. Mao, "On CSI-based vital sign monitoring using commodity WiFi," *ACM Transactions on Computing for Healthcare*, vol. 1, no. 3, pp. 1–27, 2020.

[7] E. Horvitz and D. Mulligan, "Data, privacy, and the greater good," *Science*, vol. 349, no. 6245, pp. 253–255, 2015.

[8] Y. Liu, L. Xu, X. Yuan, C. Wang, and B. Li, "The right to be forgotten in federated learning: An efficient realization with rapid retraining," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1749–1758.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, ser. Proceedings of Machine Learning Research, vol. 54. PMLR, 2017, pp. 1273–1282.

[10] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[11] L. Shen, Y. Zhang, J. Wang, and G. Bai, "Better together: Attaining the triad of byzantine-robust federated learning via local update amplification," in *Proceedings of the 38th Annual Computer Security Applications Conference*, ser. ACSAC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 201–213.

[12] M. Fang, J. Liu, N. Z. Gong, and E. S. Bentley, "Aflguard: Byzantine-robust asynchronous federated learning," in *Proceedings of the 38th Annual Computer Security Applications Conference*, ser. ACSAC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 632–646.

[13] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*. USENIX Association, 2020, pp. 1605–1622.

[14] V. Smith, C. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 2017, pp. 4424–4434.

[15] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 18–24 Jul 2021, pp. 6357–6368.

[16] H.-Y. Chen and W.-L. Chao, "On bridging generic and personalized federated learning for image classification," in *International Conference on Learning Representations*. OpenReview.net, 2022.

[17] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Álvarez, "Personalized federated learning with first order model optimization," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[18] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society, 2021.

[19] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, "Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 2545–2555.

[20] D. Yin, Y. Chen, K. Ramchandran, and P. L. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, ser. Proceedings of Machine Learning Research, vol. 80. PMLR, 2018, pp. 5636–5645.

[21] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, vol. 30. Curran Associates, Inc., 2017, pp. 119–129.

[22] F. Sattler, K. Müller, T. Wiegand, and W. Samek, "On the byzantine robustness of clustered federated learning," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2020, pp. 8861–8865.

[23] J. Xu, S. Huang, L. Song, and T. Lan, "Byzantine-robust federated learning through collaborative malicious gradient filtering," in *42nd IEEE International Conference on Distributed Computing Systems, ICDCS 2022, Bologna, Italy, July 10-13, 2022*. IEEE, 2022, pp. 1223–1235.

[24] S. P. Karimireddy, L. He, and M. Jaggi, "Byzantine-robust learning on heterogeneous datasets via bucketing," in *International Conference on Learning Representations*, 2022.

[25] Y. Liu, C. Chen, L. Lyu, F. Wu, S. Wu, and G. Chen, "Byzantine-robust learning on heterogeneous data via gradient splitting," in *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 2023, pp. 21 404–21 425.

[26] M. Baruch, G. Baruch, and Y. Goldberg, *A Little is Enough: Circumventing Defenses for Distributed Learning*. Red Hook, NY, USA: Curran Associates Inc., 2019, vol. 32.

[27] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society, 2021.

[28] C. Xie, O. Koyejo, and I. Gupta, "Fall of empires: Breaking byzantine-tolerant SGD by inner product manipulation," in *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*, ser. Proceedings of Machine Learning Research, vol. 115. AUAI Press, 2019, pp. 261–270.

[29] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems*, vol. 2. mlsys.org, 2020, pp. 429–450.

[30] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 13–18 Jul 2020, pp. 5132–5143.

[31] Y. Dai, Z. Chen, J. Li, S. Heinecke, L. Sun, and R. Xu, "Tackling data heterogeneity in federated learning with class prototypes," in *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*. AAAI Press, 2023, pp. 7314–7322.

[32] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Fedavg with fine tuning: Local updates lead to representation learning," vol. 35, pp. 10 572–10 586, 2022.

[33] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," in *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*. IEEE, 2020, pp. 1–9.

[34] S. Han, B. Buyukates, Z. Hu, H. Jin, W. Jin, L. Sun, X. Wang, W. Wu, C. Xie, Y. Yao, K. Zhang, Q. Zhang, Y. Zhang, C. Joe-Wong, S. Avestimehr, and C. He, "Fedsecurity: A benchmark for attacks and defenses in federated learning and federated llms," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*. ACM, 2024, pp. 5070–5081.

[35] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 35, no. 7, pp. 8726–8746, 2024.

[36] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *IEEE Trans. Signal Process.*, vol. 70, pp. 1142–1154, 2022.

[37] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, ser. Proceedings of Machine Learning Research, vol. 80. PMLR, 2018, pp. 3518–3527.

[38] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009.

[39] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 2921–2926.

[40] Z. Allen-Zhu, F. Ebrahimianghazani, J. Li, and D. Alistarh, "Byzantine-resilient non-convex stochastic gradient descent," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[41] J. R. Kwapisz, G. M. Weiss, and S. Moore, "Activity recognition using cell phone accelerometers," *SIGKDD Explor.*, vol. 12, no. 2, pp. 74–82, 2010.

[42] P. Kairouz, H. B. McMahan, B. Avent *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[43] 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 *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 9, 2021, pp. 7865–7873.

[44] S. Li, E. C. H. Ngai, and T. Voigt, "An experimental study of byzantine-robust aggregation schemes in federated learning," *IEEE Transactions on Big Data*, pp. 1–13, 2023.

[45] J. Oh, S. Kim, and S.-Y. Yun, "FedBABU: Toward enhanced representation for federated image classification," in *International Conference on Learning Representations*, 2022.

Appendix A
System Scalability, Overhead, and Efficiency

A. Scalability

In read-world deployments, FedCAP not only customizes models to meet participants' personalized needs but is also scalable for future clients. For example, when a future client i requests the model customization service, the server sends the global model $\boldsymbol{w}^{T-1}$ to client i. Upon receiving the model, client i conducts local updating and uploads the model update $\boldsymbol{d}_i$ to the server. Afterward, the server customizes the model $\hat{\boldsymbol{w}}_i$ using the uploaded model update $\boldsymbol{d}_i$, the calibrated update pool $\{\tilde{\boldsymbol{d}}_k^{T-1}\}_{k\in S_{T-1}}$, and the recovered model pool $\{\tilde{\boldsymbol{w}}_k^{T-1}\}_{k\in S_{T-1}}$ through Section V-A. Once client i receives the customized model $\hat{\boldsymbol{w}}_i$, it can directly perform model inference or further fine-tuning.

B. Overhead

FedCAP proposes three modules: model customization, personalized training, and model update calibration.

1) Computational Overhead: During customized aggregation, the server only needs to compute cosine similarity among model updates of N participating clients to determine the customized aggregation weights, where typically $N << K$ (the total number of clients). In the personalized training phase, although clients need to alternately update personalized models and customized models within the same mini-batch, results from ablation experiments in Section VI-E suggest that customized models already achieve satisfactory model performance. In the model update calibration phase, the computational overhead introduced is negligible compared to model customization. Although the computational complexity of existing AGRs (e.g., Median and RFA) can be reduced to $O(K)$, their combination with non-IID defenses (i.e., Bucketing and GAS) or personalized FL methods (e.g., Ditto) still cannot ensure robustness across various attack scenarios in non-IID settings (see Section VI-B4 and Section VI-B3). Our work balances system overhead with robustness against both data heterogeneity and Byzantine attacks. Additionally, FedCAP is orthogonal to computation-efficient methods. For instance, since the classification layer of the model contains more personalized features [16], [45], future work will explore calculating similarity only for the last layer.

2) Communication Overhead: Compared to the client-side model customization method FedFomo, where the server needs to send multiple models to clients in each communication round, FedCAP only needs to send one customized model to each client, resulting in communication overhead equivalent to FedAvg.

3) Storage Overhead: FedCAP only requires storing model pools and model update pools for the N participating clients. In real-world scenarios, central servers often have abundant storage resources [42], making storage overhead acceptable. Additionally, FedCAP can be combined with storage-efficient methods such as model quantization to further improve system efficiency.

In summary, considering the improvement in robustness against both data heterogeneity and Byzantine attacks provided by FedCAP, as well as the unsatisfactory performance improvement of the combination of robust FL and personalized FL methods, the introduced overhead is acceptable.

C. Efficiency Analysis

TABLE V
System Efficiency Analysis of FedCAP

Method	R2Acc(80%)	Computation(clients)	Computation(server)	Communication
FedAvg-FT	52th	3.2min	0.2s	12MB / round
FedRoD	34th	4.0min	0.2s	12MB / round
FedFomo	11th	2.6min	0.9s	120MB / round
Ditto	44th	6.2min	0.2s	12MB / round
FedCAP	9th	6.2min	2.6s*	12MB / round

*customized aggregation: 2.47s, calibration: 0.10s, detection: 0.07s

To compare FedCAP's system efficiency with other baselines, we report their Round-to-Accuracy (R2Acc), breakdown of computation time on both server and client sides, and communication overhead in the benign scenario using CIFAR-10. The R2Acc represents the number of rounds required to achieve a target accuracy (i.e., 80% on CIFAR-10), which reflects the convergence speed of FL algorithms. The communication overhead involves the data volume of models transmitted back and forth between the server and clients, and its value is determined by the number and size of models transmitted each round.

As shown in Table V, FedCAP achieves the target accuracy in the 9th round, which is on average 4 times that of others. Although FedFomo shows comparable R2Acc (11th round), its communication overhead is about 10 times that of FedCAP, as its server needs to send multiple models (10 for CIFAR-10) to each client per round to compute aggregation weights. The client-side computation times of FedRoD, Ditto, and FedCAP are about 1.3, 1.9, and 1.9 times that of FedAvg+FT, respectively, because clients need to perform additional computations for training personalized heads or models. Moreover, although FedCAP's server-side components such as customized aggregation consume additional time compared to average aggregation, this is acceptable since FedCAP's R2Acc is significantly better than that of the others. This indicates that FedCAP can reduce the overall computational overhead by decreasing the number of FL training rounds (due to its high convergence speed), thereby improving system efficiency.

In terms of scalability, FedCAP has the same communication overhead as FedAvg, increasing with the complexity of the transmitted model. As the dataset size grows, although FedCAP's personalized training introduces additional computation overhead compared to FedAvg, its customized models have achieved high accuracy (see Table IV-③, ⑤). Therefore, in read-world deployments, a trade-off between model performance and system efficiency can be achieved by selecting between customized or personalized models.