Defending the Edge: Representative-Attention for Mitigating Backdoor Attacks in Federated Learning

Chibueze Peace Obioma[1], Youcheng Sun[1,2], and Mustafa A. Mustafa[1,2]

The University of Manchester, Manchester, UK
COSIC, KU Leuven, Leuven, Belgium
`{chibueze.obioma,youcheng.sun,mustafa.mustafa}@manchester.ac.uk`

Abstract. Federated learning (FL) enhances privacy and reduces communication cost for resource-constrained edge clients by supporting distributed model training at the edge. However, the heterogeneous nature of such devices produces diverse, non-independent, and identically distributed (non-IID) data, making the detection of backdoor attacks more challenging. In this paper, we propose a novel federated representative-attention-based defense mechanism, named FeRA, that leverages cross-client attention over internal feature representations to distinguish benign from malicious clients. FeRA computes an anomaly score based on representation reconstruction errors, effectively identifying clients whose internal activations significantly deviate from the group consensus. Our evaluation demonstrates FeRA's robustness across various FL scenarios, including challenging non-IID data distributions typical of edge devices. Experimental results show that it effectively reduces backdoor attack success rates while maintaining high accuracy on the main task. The method is model-agnostic, attack-agnostic, and does not require labeled reference data, making it well suited to heterogeneous and resource-limited edge deployments.

Keywords: Federated Learning · Edge devices · Backdoor Attack.

1 Introduction

Federated learning (FL) enables a set of distributed clients to collaboratively train a global model without sharing raw data [11]. In FL, edge devices train local models on their user data and periodically send updates to a central server which aggregates them. While preserving data privacy, FL introduces new attack surfaces. An especially subtle and harmful threat is the backdoor attack, which is a form of targeted model poisoning attack [2]. In a backdoor attack, malicious clients inject a hidden behavior into the global model: the model performs normally on standard (clean) inputs, but misclassifies inputs that contain a specific attacker-chosen trigger into a target class. Since backdoored models maintain high accuracy on benign data, detecting them is difficult [1]. Unlike

untargeted attacks that noticeably degrade overall accuracy, backdoors remain stealthy until the trigger is present, thus requiring specialized defense strategies.

Defenses against backdoor attacks in FL are generally categorized into three groups: pre-aggregation, in-aggregation, and post-aggregation defenses [18]. Pre-aggregation methods, such as Krum [3], Bulyan [12], AFA [14], and Auror [20], aim to detect and exclude malicious client updates before aggregation. Anomaly detection techniques [17] also fall into this category. However, these approaches often rely on strong assumptions about data distributions and attack patterns, which may not hold in practice, making them vulnerable to sophisticated or co-ordinated backdoor attacks [18]. In-aggregation strategies, like differential privacy [15] and robust learning rates [19], mitigate threats during model aggregation without specific assumptions about attack methodologies. While effective against certain backdoor attacks, they typically degrade the global model's accuracy, presenting a trade-off between robustness and performance [23]. Post-aggregation defenses focus on restoring model integrity after aggregation through techniques like model pruning and fine-tuning [4]; however, their effectiveness against complex semantic backdoors remains uncertain [18]. Thus, there is a need for a defense that is robust, does not degrade main-task accuracy and introduces minimal overhead, especially for resource-constrained edge clients.

Recent research [24] suggests that analyzing the internal feature representations of models can reveal backdoor anomalies that are not evident from weights alone. When all clients are benign and learning the same task, their intermediate neural representations tend to align in a common feature space. In contrast, a backdoored model, despite behaving normally on clean data, often develops divergent internal activations due to optimizing for the trigger behavior.

In this paper, we propose a federated representative-attention-based (FeRA) mechanism which is a novel anomaly detection method that leverages an attention mechanism over client representations to identify malicious updates. Our approach requires only a small set of common reference inputs and does not assume any prior knowledge of the attack. In summary, our contributions are:

- We propose FeRA, a novel FL defense leveraging self-attention on intermediate model activations. By reconstructing client representations from others, our method highlights clients that deviate from the global feature space.
- Unlike prior methods that use fixed metrics and analyzes client-submitted weight vectors making it incompatible to secure aggregation, our approach captures complex relationships across clients' models, allowing a more adaptive and robust comparison than simple pairwise similarities.
- We evaluate the robustness of FeRA against state-of-the-art (SOTA) backdoor attacks under diverse non-IID federated learning conditions. Our results show that FeRA significantly reduces backdoor success rates while preserving clean model performance and incurring minimal overhead on resource-constrained edge clients.

The rest of the paper is organized as follows. Section 2 presents background information. Section 3 introduces our proposed solution - FeRA. In Section 4,

we provide an extensive evaluation of FeRA. Section 5 surveys related work. Section 6 concludes the paper and outlines potential future directions.

2   Background and Problem Setting

In this section, we provide a detailed background on FL, backdoor attacks in FL, and introduce key ideas underpinning attention-based model interpretability.

FL in Edge Environments. FL has emerged as a key paradigm for privacy-preserving collaborative model training across distributed, resource-constrained edge devices.

Let $\{D_k\}_{k=1}^K$ be the datasets stored across K edge devices (or *clients*), each training locally without exposing raw data. Formally, FL seeks to solve:

$$\min_{\mathbf{w}} \ F(\mathbf{w}) \ = \ \sum_{k=1}^{K} \frac{|D_k|}{|D|} \, F_k(\mathbf{w}), \quad F_k(\mathbf{w}) \ = \ \frac{1}{|D_k|} \sum_{(\mathbf{x}_i,\, y_i) \in D_k} \ell\big(\mathbf{w}; \mathbf{x}_i, y_i\big), \quad (1)$$

where $\ell(\cdot)$ is the local loss, $\mathbf{w}$ is the model parameters, and $|D| = \sum_{k=1}^K |D_k|$ [11]. The server initiates a global model $\mathbf{w}^{(0)}$ and repeatedly updates it: each device k computes $\mathbf{w}_k^{(t)} = \mathbf{w}^{(t-1)} - \eta \, \nabla F_k\big(\mathbf{w}^{(t-1)}\big)$ and the server aggregates via $\mathbf{w}^{(t)} = \sum_{k=1}^{m} \alpha_k \, \mathbf{w}_k^{(t)}$, where η is the local learning rate, and $\alpha_k = |D_k| \, / \, \sum_{j=1}^{m} |D_j|$. Only a fraction m of devices (out of K) may be selected each round, reflecting limited communication or power constraints in edge deployments. Edge devices often operate autonomously and may be physically exposed or weakly secured, increasing their susceptibility to attacks and amplifying the risk of targeted model poisoning attacks [18]. Thus, methods for securing FL systems, especially under non–IID data, remain an active research area.

Backdoor attacks in FL. Backdoor attacks are a targeted poisoning strategy wherein malicious clients attempt to implant a specific misclassification behavior (the *backdoor*) into the global model [2]. Concretely, an adversary chooses a trigger T (e.g., a pixel pattern, or a rare input sample for semantic backdoors) and a target label y_t. During local training, the malicious client injects samples $\big(\mathbf{x}+T, \, y_t\big)$ into its dataset. The goal is for the global model to learn $f_{\mathbf{w}^{(*)}}(\mathbf{x}+T) = y_t$, while still retaining high accuracy on clean inputs $\mathbf{x}$.

Since FL only inspects local updates, poison samples remain concealed [21]. Classic centralized backdoor attacks (e.g., BadNets [7]) were extended to FL by boosting malicious updates to overcome the diluting effect of aggregation. FL backdoor variants appear in different forms. For instance, *model poisoning* [1] lets attackers rescale or directly manipulate local gradients ∇F_k to dominate the aggregation; *edge-case backdoors* [25] exploit rare or out-of-distribution "edge" inputs to insert subtle targeted attacks; and *distributed backdoor attacks* (DBA) [27] split a trigger across multiple colluding clients, making each update appear benign while collectively embedding the backdoor. Mathematically, these strategies exploit the FL averaging operation to push the global parameters $\mathbf{w}^{(t)}$

toward a poisoned optimum $\mathbf{w}^{(*)}$. The result is a model that performs normally on clean data but consistently predicts y_t on triggered inputs $\mathbf{x} + T$.

Representation via attention While most defenses focus on weigh vectors or gradient norms, recent studies reveal that internal feature representations offer more discriminative cues for distinguishing malicious from benign model behavior [13,24]. Specifically, if $\phi_{\mathbf{w}}(\mathbf{x})$ denotes the intermediate-layer activations of a model with parameters $\mathbf{w}$, then benign clients tend to exhibit similar high-level feature representations on clean reference inputs, while backdoored models show anomalous deviations.

Previous approaches such as ARIBA [13] analyze static model parameters, e.g., convolutional kernel gradients - breaking them into smaller fragments and applying unsupervised outlier scoring (e.g., Mahalanobis distance) to detect structural bias caused by backdoors. FLDetector [28], on the other hand, employs consistency in client updates across rounds, using the Cauchy mean value theorem to predict expected behavior and identify attackers who deviate from it. These approaches either require hand-crafted statistical tests or rely on multi-round histories, making them less flexible for real-time, round-level detection.

More closely aligned with our method, FedAvgCKA [24] introduced the use of centered kernel alignment (CKA) to quantify pairwise similarity between client feature representations on a shared root dataset. Malicious models are flagged based on their low similarity with the majority. However, computing all pair-wise similarities across multiple layers incurs high overhead. In contrast, our proposed defense adapts the multi-head self-attention mechanism introduced in the Transformer architecture by Vaswani et al. [22] to the federated setting for client model comparison. Instead of using attention to aggregate token-level sequences, we apply it over client-level representation vectors derived from shared reference inputs. Each client's representation is treated as a query and reconstructed via attention-weighted combinations of its peers' representations. The core idea is that benign clients, sharing similar internal representations, can effectively "explain" each other, yielding low reconstruction error. In contrast, a malicious client whose representation deviates due to backdoor objectives cannot be accurately reconstructed from others, resulting in a high anomaly score.

This attention-based mechanism allows us to move beyond static similarity metrics (e.g., cosine distance, kernel alignment), to capturing more nuanced dynamic relationships across models. Unlike CKA-based defenses that rely on fixed pairwise statistics or ARIBA's handcrafted outlier tests on filter weights, FeRA introduces a learned, adaptive framework for anomaly detection. Each attention head can capture different subspaces of representation similarity, enabling richer modeling of inter-client behavior. Thus, FeRA provides a flexible and generalizable approach to identifying poisoned models even in heterogeneous FL systems.

System model. We consider a FL system typical of edge deployments, consisting of a central server and a set of K client devices (edge nodes). The clients collaboratively train a global model (e.g., a deep neural network for image or sensor data classification). In each training round t, a subset of m clients (out

of K) is selected and receives the current global model parameters $w^{(t)}$. Each selected client k uses its local dataset D_k to perform some training (e.g., a few epochs of stochastic gradient descent (SGD) and obtains an updated model $w_k^{(t)}$. The client then sends this update (or the model parameters) to the server. The server aggregates the m received models into a new global model $w^{(t+1)}$. This process repeats for T rounds until convergence.

Adversary goals and adversarial capabilities. In line with federated backdoor literature [16,5,27], we consider a *honest-but-curious* aggregator that faithfully performs model aggregation but lacks direct visibility into client data or any knowledge of the attacker's backdoor trigger patterns. A minority subset of clients is compromised, denoted by $\mathcal{M}$ with $|\mathcal{M}| = m < \frac{K}{2}$. These malicious devices can perform *data poisoning* (e.g., flipping labels or inserting trigger samples $(x \oplus T, y_{\mathrm{mal}})$) and/or *model poisoning* (e.g., scaling or constraining local updates) to embed a covert mapping $x \oplus T \mapsto y_{\mathrm{mal}}$ into the global model parameters $\mathbf{w}^{(t+1)}$. They strive to remain close to benign updates on non-trigger data so as to evade detection, potentially colluding on their poisoning strategies to inject or distribute backdoors more effectively. Following the model in [16,27], the adversary can adapt its malicious contributions if it anticipates the server's defense. Attackers, however, cannot alter the server's aggregation routine or the reference data used to evaluate internal representations. For edge devices, an overt drop in main-task accuracy or large parameter divergence would immediately raise suspicion. Thus, malicious clients often resort to partial attacks over multiple rounds or carefully tune their gradients to approximate benign signals.

Defense Objective. Our main objective is to neutralize any backdoor functionality while preserving the global model's accuracy on benign inputs. Specifically, we seek to constrain malicious updates so that no persistent mapping $x \oplus T \mapsto y_{\mathrm{mal}}$ can take root in the aggregated model. By leveraging cross-client *representation consistency* checks, our defense aims to detect and exclude anomalous activations before they significantly bias model convergence toward the trigger. In doing so, we ensure that attackers cannot dominate learned feature subspaces even when they collectively attempt to manipulate their updates.

3   FeRA Overview and Design

Unlike prior methods [24,13] that use fixed metrics (e.g., cosine distance, kernel alignment) to compare models, FeRA employs a self-attention module to reconstruct each client's representation from those of other clients (see Fig. 1). By operating at the representation level, FeRA is resilient to weight obfuscation or scaling attacks, focusing instead on functional behavior. Moreover, the attention mechanism captures complex relationships across clients' models, allowing a more adaptive and robust comparison than simple pairwise similarities. This makes our method especially suited for edge devices, where non-IID data is the norm and adversaries can be stealthy and adaptive.

Fig. 1. A high-level overview of FeRA: at each round, the server performs representation extraction and attention-based anomaly detection before aggregating models.

Overview. In each training round t, the server performs the following steps: (1) It receives updated local models $w_1, w_2, \ldots, w_m$ from the m selected clients (after each client trains on its local data for that round). (2) It feeds a small reference dataset D_{ref} into each client's model to obtain internal activations at a chosen layer. These activations are aggregated to form a representative feature vector r_i for each client's model W_i. (3) Using a multi-head self-attention mechanism, it attempts to reconstruct each client's representative vector using the vectors from all other clients. This produces a reconstructed representation $\hat{r}_i$ for each client i based on its peers' representations. (4) It computes the reconstruction error for each client i, defined as the distance between r_i and its reconstruction $\hat{r}_i$ ($e_i = \|r_i - \hat{r}_i\|_2$). This scalar e_i serves as an anomaly score indicating how well client i's update aligns with the others. (5) It converts each anomaly score into a non-negative weight, assigning higher weights to clients with low scores and zero weight to those exceeding the threshold.

Design Challenges. We outline three core design challenges encountered.

Ch-1 - Selection of representation layer. Choosing the appropriate neural layer from which to extract features is critical. If the chosen layer is too early in the network, it will capture mostly low-level or generic features that are common to all inputs, making it hard to distinguish subtle backdoor-induced anomalies. On the other hand, if the representation is taken from a layer too close to the output, an adversary could manipulate the model's final layers to hide the backdoor influence, thereby masking internal anomalies. We balance this trade-off by selecting an intermediate-high layer that captures high-level semantic features yet is not overly influenced by the final classifier. We found that using the penultimate fully-connected layer for CNN models (or the global pooling layer in residual architectures) yields representations that are both discriminative and

stable across clients. We validated this choice through ablation studies in Appendix B, observing that backdoored models tend to produce markedly divergent embeddings at these layers while benign models cluster tightly in feature space.

Ch-2 - Non-IID diversity. In federated edge systems, clients naturally operate on heterogeneous and non-IID data distributions. This non-IID nature poses a challenge for anomaly detection, as benign clients may produce divergent representations that mimic the behavior of malicious updates. Traditional distance-based methods often misclassify such benign deviations as attacks, especially under severe heterogeneity. FeRA mitigates this issue using a self-attention-based reconstruction strategy, where each client's embedding is evaluated relative to others. This relational encoding allows FeRA to tolerate diverse but benign client behavior while identifying truly anomalous (and often non-reconstructable) representations. We evaluate this capability under varying Dirichlet non-IID settings in §4, showing that FeRA maintains high detection fidelity even as data distributions diverge.

Ch-3 - Ensuring computational efficiency and scalability. Incorporating an attention based detection mechanism into each training round introduces additional overhead that must be kept feasible for large-scale deployments. A naive implementation of multi-head attention across m clients would incur $O(m^2 \cdot d)$ time complexity for d-dimensional representations, which can be prohibitive as m grows. We employ several techniques to ensure scalability: we limit the number of clients processed per round to a fixed-size subset when necessary (many FL systems already sample a subset of clients each round); we restrict the reference dataset D_{ref} to a small but representative set of inputs, which keeps the cost of forward passes and the dimension of each representation manageable; we implement the attention operations in a vectorized manner and optimize the dimensionality d of the chosen layer's embedding (using a pooling layer that yields a compact vector) to reduce computation without sacrificing fidelity. Our analysis in §4 confirms that the overall FeRA runtime grows modestly with clients number.

FeRA Design. We formalize FeRA in Alg. 1. This algorithm is integrated into the FL workflow at the server side. Next, we describe each component in detail.

Representation extraction. The first component of FeRA aims to obtain a comparable feature representation from each client's model. Prior work [24] suggests that backdoor differences often manifest in deeper layers, so ℓ could be the penultimate layer or last convolutional layer of a CNN. We designate a particular internal layer ℓ of the neural network (as determined by the analysis in C1) to extract activations. Formally, let $f_i^{(\ell)}(x)$ denote the output feature vector of model W_i at layer ℓ for input x. The server uses the common reference dataset $D_{\mathrm{ref}} = x^{(1)}, x^{(2)}, \ldots, x^{(R)}$ and computes $f_i^{(\ell)}(x^{(j)})$ for each client i and each sample $x^{(j)} \in D_{\mathrm{ref}}$. This yields a set of R activation vectors per client. We then condense these into a single representative embedding for each client. A simple

---

Algorithm 1 FeRA: Federated Representative-Attention

Require: global model G_{t-1}; clients $\{C_1, \dots, C_m\}$; reference set $\mathcal{D}_{\text{ref}}$; threshold $\tau_{|z|}$
Ensure: updated global model G_t
 1: **for** each client C_i **do**   ▷ local update
 2:     $W_i \leftarrow \text{LocalTrain}(G_{t-1})$
 3: **end for**
 4: **for** each W_i **do**   ▷ representations
 5:     $r_i \leftarrow \frac{1}{|\mathcal{D}_{\text{ref}}|} \sum_{\mathbf{x}} f_{W_i}^{(\ell)}(\mathbf{x})$
 6: **end for**
 7: **for** each client C_i **do**   ▷ attention reconstruction
 8:     $\hat{r}_i \leftarrow \text{AttentionReconstruction}(r_i, R)$
 9:     $e_i \leftarrow \| r_i - \hat{r}_i \|_2$
10: **end for**
11: $\tilde{e} \leftarrow \text{median}\{e_i\}, \quad \text{MAD} \leftarrow \text{median}\,|e_i - \tilde{e}| + \varepsilon$
12: **for** each e_i **do**   ▷ robust anomaly score
13:     $|z_i| \leftarrow 0.6745\,|e_i - \tilde{e}|/\text{MAD}$
14: **end for**
15: **for** each client C_i **do**   ▷ compute soft aggregation weights
16:     $\omega_i \leftarrow \max\{0, \tau_{|z|} - |z_i|\}$
17: **end for**
18: normalize ω_i: $\omega_i \leftarrow \omega_i \Big/ \sum_{j=1}^{m} \omega_j$
19: $G_t \leftarrow \sum_{i=1}^{m} \omega_i\, W_i$
20: **return** G_t

---

and effective choice is to take the average activation:

$$r_i = \frac{1}{R} \sum_{j=1}^{R} f^{(\ell)} i(x^{(j)}) \tag{2}$$

where $r_i \in \mathbb{R}^d$ (if layer ℓ has dimension d). This r_i serves as client C_i's semantic fingerprint on the reference data, summarizing how model W_i internally represents typical inputs. All representation extraction occurs on the server using the received models W_i; D_{ref} need not be labeled, and we assume it does not contain any adversarial trigger pattern. At the end of this stage, the server has a matrix $R = r_1, \dots, r_m$ of size $m \times d$, containing one embedding per client model.

Attention-based reconstruction mechanism. The core novelty of FeRA lies in determining whether each client's representation r_i can be reconstructed from the representations of other clients. Intuitively, if all clients are training on similar data without malicious interference, their feature representations should reside in a common latent space and be mutually predictive [24]. We operationalize this intuition using a multi-head self-attention mechanism [22]. FeRA's attention module treats the target embedding r_i as a query and the set of other client embeddings $r_j : j \neq i$ as a collection of keys and values in an attention operation. Algorithm 2 details this reconstruction procedure. We learn projection

---

Algorithm 2 AttentionReconstruction(r_i, R)

Require: Target embedding $r_i \in \mathbb{R}^d$; set of embeddings $R = \{r_1, \ldots, r_m\}$
Ensure: Reconstructed embedding $\hat{r}_i$
1: **for** each client $r_j \in R$ **do**
2:     Compute projections:

$$q_i = r_i W^Q, \quad k_j = r_j W^K, \quad v_j = r_j W^V$$

3:     Compute attention score:

$$e_{ij} = \exp\left(\frac{q_i^\top k_j}{\sqrt{d}}\right)$$

4: **end for**
5: Normalize attention weights:

$$\alpha_{ij} = \frac{e_{ij}}{\sum_{j'} e_{ij'}}$$

6: Compute reconstruction:

$$\hat{r}_i = \sum_j \alpha_{ij} v_j$$

7: **return** $\hat{r}_i$

---

matrices W^Q, W^K, W^V that map each vector r into a query vector q, a key vector k, and a value vector v (all in $\mathbb{R}^d$). A detailed formulation is provided in Appendix A. For client i, its query q_i is used to compute attention weights against the keys k_j of every other client j, via a scaled dot-product and softmax normalization (Lines 2–5 in Alg. 2). These weights α_{ij} indicate how much client i's representation aligns with each other client j's features. We then produce the reconstructed vector $\hat{r}_i$ as a weighted combination of the value vectors v_j from all other models (Line 6 of Alg. 2). As we employ multiple attention heads (with independent projections for each head), this mechanism can capture diverse facets of representation similarity (each head attending to different subspaces of the features). The result $\hat{r}_i$ is the best approximation of r_i that can be formed using the information from the other $m - 1$ clients in that round.

After computing $\hat{r}_i$ for each client, FeRA measures how well this reconstruction matches the client's actual representation r_i using the Euclidean distance $e_i = |r_i - \hat{r}_i|_2$. To robustly detect anomalies, we calculate the median and Median Absolute Deviation (MAD) of these errors across clients:

$$\tilde{e} = \text{median}(e_i), \quad \text{MAD} = \text{median}(|e_i - \tilde{e}|) + \varepsilon \tag{3}$$

We then compute a robust anomaly score via a modified two-tailed z-score:

$$|z_i| = 0.6745 \frac{|e_i - \tilde{e}|}{\text{MAD}} \tag{4}$$

This approach flags clients whose errors significantly deviate from the median, regardless of direction. Clients with anomaly scores exceeding a threshold $\tau_{|z|}$

are assigned zero weight in the aggregation, while others' updates are weighted proportionally to their proximity to the threshold. In practice, $\tau_{|z|}$ is tuned on a small validation set of known-clean models. Finally, FeRA down-weights suspicious clients based on anomaly scores during aggregation, mitigating backdoor effects without fully excluding updates, thus preserving robustness even under client noise.

4 Experimental Evaluation

Setup. Below we provide details of our experimental setup.

System configuration and baselines. We simulate an FL system with 100 clients. In each communication round, 20% of clients are randomly selected for participation. The proportion of malicious clients is varied 0-50%. Models are trained using PyTorch (v2.0) for 500 communication rounds. We compare FeRA with a suite of established baselines: Multi-Krum [3], FoolsGold [6], FLAME [16], and FLTrust [5]. We also include anomaly detection-based methods: ARIBA [13], FLDetector [28], and FedAvgCKA [24], which leverage statistical, clustering, or representational similarity techniques. Standard FedAvg [11] is used as an insecure baseline. All defenses are implemented and tuned according to their specs.

Datasets and models. We evaluate FeRA on three image classification datasets: MNIST [9] (70,000 28×28 grayscale, 10 classes), F-MNIST [26] (70,000 28×28 grayscale images, 10 classes), and CIFAR-10 [8] (60,000 32×32 color images, 10 classes). Data is partitioned across clients using a Dirichlet distribution to simulate moderate non-i.i.d. conditions. Local models consist of a 4-layer CNN for MNIST and F-MNIST. For CIFAR-10, we use a ResNet-18 architecture. All models are identically initialized to isolate the impact of malicious updates.

Backdoor attack scenarios. We evaluate FeRA against SOTA backdoor attacks: DBA [27], Edge-Case backdoor [25], and model poisoning attack [1]. In all cases, poisoned examples are *label-consistent* (i.e., labeled with the attacker's target class), and attackers begin injecting poisons from their first participation.

Evaluation metrics. We evaluate each defense using four key metrics. First, the attack success rate (ASR) quantifies the percentage of backdoored test inputs misclassified as the target label; lower ASR indicates stronger defense capability. Second, we measure main-task accuracy (MTA) on the clean test set, with an effective defense expected to preserve accuracy close to the baseline model trained without any attack. Finally, we assess the computational cost of each defense in terms of server-side complexity and any added burden on client devices.

Experimental evaluation. Below we provide our evaluation results.

Table 1. Backdoor attack success rate (ASR) and main-task accuracy (MTA) for FedAvg (no defense) versus FeRA on three attacks.

| Attack | Dataset | FedAvg (no defense) | | **FeRA (Ours)** | |
		ASR	MTA	ASR	MTA
DBA	MNIST	100.0	98.7	0.6	98.2
Edge case	CIFAR-10	100.0	88.0	4.0	85.7
Model poisoning	F-MNIST	100.0	97.3	2.1	96.0

Attack mitigation effectiveness. We examine FeRA's effectiveness in mitigating backdoor attacks compared to undefended FL systems. Table 1 summarizes the backdoor ASR and MTA for FedAvg (no defense) versus FeRA on three attacks. As expected, FedAvg without defense is vulnerable, with ASR $\approx 100\%$ across all cases. FeRA effectively neutralizes the backdoors, reducing ASR to 0.6% under DBA on MNIST. Similarly, FeRA lowers ASR on CIFAR-10 to 4%. For the model poisoning case on F-MNIST, FeRA reduces ASR to 2.1%, with a minimal drop in MTA from 97.3% to 96%. These results demonstrate that FeRA simultaneously delivers strong resilience against diverse backdoor strategies while sustaining high clean accuracy. Small accuracy drops are due to occasional filtering of benign updates, yet these remain acceptable given the security gains.

Comparison with baseline defenses. Table 2 presents ASR and MTA on all three datasets (MNIST, CIFAR-10, F-MNIST) for each defense method outlined in Section 4. In a comparative evaluation against DBA in FL, FeRA demonstrated superior performance across MNIST, CIFAR-10, and F-MNIST datasets. Under conditions of moderate non-i.i.d. data ($\alpha = 0.4$) and 20% adversarial clients, FeRA achieved the lowest ASR, notably 8.1% on CIFAR-10, 1.0% on MNIST, and 1.4% on F-MNIST, while maintaining high MTA close to the baseline. Although methods like ARIBA and FedAvgCK also kept ASR in low, FeRA consistently outperformed them, especially on more complex datasets like CIFAR-10.

Other defenses exhibited trade-offs between robustness and accuracy. The technique by FLAME reduced ASR to 7.3% by clipping and adding noise to updates, but this led to significant drops in MTA (12.2% decrease on CIFAR-10). FLTrust maintained a balance with low ASR and high MTA, yet FeRA exceeded its performance. FeRA's advantage is further highlighted by its rapid detection and removal of malicious clients, often within the first attack round, preventing backdoor accumulation. Overall, FeRA offers a robust defense against backdoor attacks in FL, providing strong security with minimal impact on model utility.

Impact of malicious client fraction. Table 3 reports the backdoor ASR and MTA as the attacker ratio increases 0-50% across three datasets. As the fraction of malicious clients grows, FeRA generally maintains high clean accuracy and keeps backdoor attacks to negligible levels up to moderate adversarial participation ($\leq 30\%$). For instance, at 20% compromise, backdoor ASRs remain under 2%

Table 2. Backdoor attack success rate (ASR) and main-task accuracy (MTA) on all three datasets (MNIST, CIFAR-10, F-MNIST) for each defense method under DBA with $\alpha = 0.4$, and 20% of clients adversarial. All results are in %.

Defenses	CIFAR-10		MNIST		F-MNIST	
	ASR	MTA	ASR	MTA	ASR	MTA
No Defense (FedAvg)	100.0	88.0	100.0	98.7	100.0	97.3
Multi-Krum [3]	76.4	86.5	90.5	97.0	94.1	96.1
FoolsGold [6]	87.3	87.8	94.8	98.1	92.0	97.0
FLAME [16]	7.3	75.8	2.2	92.4	2.8	90.7
FLTrust [5]	11.2	82.9	3.8	95.1	3.2	96.8
ARIBA [13]	7.4	87.2	4.6	97.5	3.8	96.0
FLDetector [28]	10.5	65.1	2.3	96.3	5.9	64.4
FedAvgCKA [24]	10.2	85.5	1.3	98.4	2.7	96.2
FeRA (Ours)	8.1	86.1	1.0	98.5	1.4	96.6

Table 3. Backdoor attack success rate (ASR) and main-task accuracy (MTA) under varying level of malicious clients, all in %.

Attackers	MNIST		CIFAR-10		F-MNIST	
	ASR	MTA	ASR	MTA	ASR	MTA
0	-	98.7	-	88.1	-	97.4
10	0.6	98.5	0.5	87.5	1.0	96.9
20	1.0	98.5	0.5	87.5	1.4	96.6
30	1.2	98.0	8.9	84.2	1.3	96.1
40	19.1	98.5	82.5	87.5	65.0	96.9
50	86.9	96.0	92.4	80.3	89.5	97.1

across all datasets while MTA drops only slightly from the no-attack baseline. Even at 40% malicious clients, FeRA significantly reduced the attack success, though the backdoor ASR begins to spike (e.g., around 50% on CIFAR-10). Once half of the participants collude adversarially, the ASR reaches 90% success on some datasets, yet FeRA still averts a complete compromise and maintains respectable model accuracy. These results affirm FeRA's resilience in all but the most extreme threat scenarios.

Adaptive feature-alignment attacker. To test FeRA against an *adaptive* adversary, we re-implemented model–poisoning with a feature-alignment loss $\mathcal{L} = \mathcal{L}_{\text{backdoor}} + \beta\mathcal{L}_{\text{clean}} + \lambda\|r_i - \bar{r}_{\text{benign}}\|_2^2$ ($\beta = 1$, $\lambda \in \{0.1, 1\}$), where the attacker minimises its MAD-z anomaly score during local Projected Gradient Descent (PGD). With 20% malicious clients and the same non-IID setting as in §4, FeRA still collapses the ASR below 10% across all the datasets and preserves MTA within reasonable bounds (Table 4), demonstrating that even an adaptive attacker cannot implant an effective back-door without being flagged by FeRA.

Table 4. FeRA versus adaptive feature-alignment back-door

Dataset	FedAvg (no defense)		FeRA	
	ASR (%)	MTA (%)	ASR (%)	MTA (%)
MNIST	100.0	98.6	**7.4**	95.4
CIFAR-10	100.0	88.0	**9.6**	81.5
Fashion-MNIST	100.0	97.2	**8.8**	90.8

Effect of data heterogeneity. We evaluate FeRA under varying degrees of client-side data heterogeneity, parameterized by the Dirichlet distribution coefficient α. As shown in Table 5, lower α values correspond to highly skewed data distributions, where each client's local dataset is dominated by one or a few classes, a setting known to severely challenge anomaly-based defenses. Despite this, FeRA consistently maintains strong defense performance across all three datasets. For example, under extreme heterogeneity ($\alpha = 0.2$), ASR improves in the whole data set compared to the results of [24]. We obtain a 2% reduction of ASR on MNIST and 3% on CIFAR-10), while MTA remains high (e.g., 98.2% and 84.3%, respectively). As α increases toward more i.i.d.-like conditions (up to $\alpha = 1.0$), FeRA's performance further improves, with ASR dropping near zero and MTA approaching its clean upper bound. These results highlight FeRA's stability and scalability in federated environments with varying data non-uniformity. FeRA's robustness is rooted in its architecture, particularly the use of a multi-head attention mechanism that reconstructs each client's latent representation by weighting contributions from other clients. This design helps mitigate the increased false positive risk typically associated with highly non-i.i.d. settings, where benign updates can diverge substantially and resemble malicious anomalies.

Table 5. FeRA's backdoor attack success rate (ASR) and main-task accuracy (MTA) (in %) under varying data heterogeneity scenarios (Dirichlet parameter α).

α	MNIST		CIFAR-10		F-MNIST	
	ASR	MTA	ASR	MTA	ASR	MTA
1.0	1.0	98.8	4.3	86.8	1.5	97.3
0.8	2.1	98.8	4.5	86.4	1.0	97.1
0.6	5.2	98.7	5.7	86.0	1.2	96.9
0.4	1.0	98.5	5.3	85.1	2.1	96.6
0.2	30.2	98.2	41.5	84.3	23.7	95.5

Influence of reference dataset size. Similar to other methods [24], [5], [10], FeRA requires a small clean reference dataset at the server to evaluate client representations and detect anomalies. To assess sensitivity to this requirement, we measure FeRA's robustness on CIFAR-10 with 10% malicious clients, varying

Fig. 2. Variation of backdoor attack success rate (ASR) with the size $|R|$ of the server's reference dataset (CIFAR-10 experiment, 10% malicious). Error bars (shaded) show the standard deviation over 3 runs.

the reference set size. Notably, FeRA achieves low backdoor ASR even with just 16 clean samples, substantially outperforming undefended baselines. Increasing the reference set size rapidly enhances detection; at 64 samples, the backdoor attack is greatly mitigated with stable results across multiple runs (see Fig. 2). Thus, FeRA remains highly effective even when trusted reference data is limited, reflecting its capability to capture rich relational cues through attention.

Computational overhead analysis. FeRA's attention-based anomaly detection introduces moderate server-side overhead, primarily from representation extraction and self-attention computations. The per-round complexity, $O(n|R|d+n^2d)$, consists of a linear term similar to FLTrust and a quadratic term comparable to Multi-Krum or FLAME, yet significantly lighter than FedAvgCKA's $O(n^2k^2d+k^3)$, providing a one-to-two order of magnitude reduction for typical CNNs. Given standard FL client sampling (10–20% per round) and a modest reference set size ($|R| = 50$–100), FeRA's overhead remains manageable. Computation is entirely server-side, imposing no additional resource demands on clients beyond standard FL ($O(d)$). Empirical results with 20 clients and $|R| = 100$ show per-round latency rising from $2\,\text{s}$ (FedAvg) to approximately $3\,\text{s}$, a $1.5\times$ increase justified by enhanced robustness; further efficiency gains are possible via dimensionality reduction, subset attention, or GPU acceleration.

Influence of poisoning ratio on FeRA. As shown in Fig. 3, FeRA demonstrates strong resilience across varying poisoning ratios. Even at a high poisoning ratio of 12.5%, FeRA constrains ASR to 26.7% on MNIST and 42.4% on CIFAR-10, while maintaining high clean accuracy. As the poisoning ratio decreases, ASR declines significantly across all datasets (e.g., dropping below 1% on MNIST and Fashion-MNIST at 25%), with MTA remaining stable. This performance reflects FeRA's architectural strength: its attention-based representation reconstruction is sensitive to even sparse malicious deviations, allowing it to scale its defense as

Table 6. Comparison of computational complexity and client memory cost. n: number of clients per round, d: model size, k: number of classes, $|R|$: reference dataset size.

Method	Server computation per round	Client memory overhead
FedAvg [11]	$\mathcal{O}(d)$	$\mathcal{O}(d)$
FedAvgCKA [24]	$\mathcal{O}(n^2 k^2 d + k^3)$	$\mathcal{O}(d)$
Multi-Krum [3]	$\mathcal{O}(n^2 d)$	$\mathcal{O}(d)$
FLAME [16]	$\mathcal{O}(n^2 d)$	$\mathcal{O}(d)$
FLTrust [5]	$\mathcal{O}(n|R|d)$	$\mathcal{O}(d)$
FeRA (Ours)	$\mathcal{O}(n\,|R|\,d + n^2 d)$	$\mathcal{O}(d)$

Fig. 3. FeRA's backdoor attack success rate (ASR) and main-task accuracy (MTA) (in %) under varying poisoning ratios.

attack severity increases. Overall, these results confirm FeRA's robustness and adaptability across a broad spectrum of adversarial conditions.

Discussion and limitations. While FeRA effectively mitigates backdoor attacks in FL, several limitations exist. First, FeRA relies on the availability of a clean reference dataset at the server. This requirement might conflict with FL's privacy principles and could be challenging in systems lacking trusted data. Second, the attention-based anomaly detection introduces considerable server-side computational overhead. This overhead, while manageable for moderate-scale deployments, raises scalability concerns in large-scale edge settings with many clients, highlighting the need for optimized or hierarchical solutions to maintain efficiency. Third, FeRA's accuracy may degrade under extreme non-IID data distributions, where benign client representations significantly diverge, increasing false positives. Moreover, adaptive adversaries capable of closely mimicking benign patterns could potentially evade detection, particularly as adversarial participation grows.

5   Related Work

Existing defenses against backdoor attacks in FL broadly focus on robust aggregation, anomaly detection, and model-level interventions. Robust aggregators,

inspired by Byzantine fault tolerance methods, seek consensus among client updates by filtering outliers, such as Krum [3] and median-based aggregation techniques. These methods assume limited malicious participation; if adversaries exceed certain bounds or mimic benign behaviors closely, robustness degrades significantly [14]. Enhancing aggregation, FLTrust [5] introduces a server-side trusted dataset to validate client updates, thereby significantly reducing backdoor risks but compromising FL's privacy constraints.

Anomaly detection methods actively identify malicious clients by analyzing statistical or representation-level discrepancies in updates. FoolsGold [6], for instance, identifies malicious collusion via cosine similarity but risks falsely penalizing benign, similarly-distributed clients. More advanced clustering-based methods, like FLAME [16], combine statistical clustering (HDBSCAN) with heuristic filtering rules, improving detection precision. Recent methods, like ARIBA[13] and FedAvgCKA [24], inspect deeper model representations or gradient distributions, effectively isolating subtle backdoor signals by assessing internal representational anomalies. Complementarily, historical consistency-based methods, exemplified by FLDetector [28], track temporal deviations in client updates, but require sustained client participation and may miss subtle, adaptive attacks.

Model-level defenses, notably pruning-based strategies like Fine-Pruning [10], directly remove neurons implicated in backdoors, effectively neutralizing attacks without extensive client monitoring. While powerful in centralized contexts, these techniques typically require substantial clean reference data and central fine-tuning, complicating their federated deployment.

Our approach, FeRA, aligns closely with representation-based anomaly detection like ARIBA and FedAvgCKA, but integrates multi-head attention mechanisms to adaptively scrutinize client updates each training round, offering improved robustness in federated settings characterized by highly non-i.i.d. distributions and resource constraints.

6   Conclusion and Future Work

We introduced FeRA, a representative-attention-based defense mechanism designed to mitigate backdoor attacks in FL. FeRA employs multi-head attention to dynamically evaluate client updates, significantly reducing backdoor attack success rates while maintaining high accuracy on legitimate tasks. Notably, FeRA achieves superior performance compared to existing defenses such as ARIBA and FedAvgCKA, particularly in challenging non-i.i.d. data scenarios, with minimal computational overhead on edge devices. Future work includes developing class-aware anomaly detection to improve fine-grained sensitivity, adaptive thresholding mechanisms for dynamic adjustment to evolving data distributions, and extending FeRA to asynchronous FL settings.

References

1. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: International conference on artificial intelligence and statistics. pp. 2938–2948. PMLR (2020)
2. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: Proceedings of the 36th International Conference on Machine Learning (ICML). pp. 634–643 (2019)
3. Blanchard, P., Mhamdi, E.M.E., Guerraoui, R., Stainer, J.: Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 118–128 (2017)
4. Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C.A., Jia, H., Travers, A., Zhang, B., Lie, D., Papernot, N.: Machine unlearning. In: 2021 IEEE symposium on security and privacy (SP). pp. 141–159. IEEE (2021)
5. Cao, X., Fang, M., Liu, J., Gong, N.: FLTrust: Byzantine-robust federated learning via trust bootstrapping. In: Network and Distributed Systems Security (NDSS) Symposium. pp. 1–15 (2021)
6. Fung, C., Yoon, C.J., Beschastnikh, I.: The limitations of federated learning in sybil settings. In: 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020). pp. 301–316 (2020)
7. Gu, T., Dolan-Gavitt, B., Garg, S.: Badnets: Identifying vulnerabilities in the machine learning model supply chain. In: Workshop on Artificial Intelligence and Security (2017), `https://arxiv.org/abs/1708.06733`, preprint arXiv:1708.06733
8. Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report, University of Toronto (2009), `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`, accessed: 2025-04-14
9. LeCun, Y., Cortes, C., Burges, C.J.C.: MNIST handwritten digit database. `http://yann.lecun.com/exdb/mnist` (2010), accessed: 2025-04-14
10. Liu, K., Dolan-Gavitt, B., Garg, S.: Fine-pruning: Defending against backdooring attacks on deep neural networks. In: International symposium on research in attacks, intrusions, and defenses. pp. 273–294. Springer (2018)
11. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
12. Mhamdi, E.M.E., Guerraoui, R., Rouault, S.: The Hidden Vulnerability of Distributed Learning in Byzantium. In: Proceedings of the 35th International Conference on Machine Learning. pp. 3518–3527 (2018)
13. Mi, Y., Guan, J., Zhou, S.: Ariba: Towards accurate and robust identification of backdoor attacks in federated learning. arXiv preprint arXiv:2202.04311 (2022)
14. Munoz-Gonzalez, L., Co, K.T., Lupu, E.C.: Byzantine-robust federated machine learning through adaptive model averaging. Preprint arXiv:1909.05125 (2019)
15. Naseri, M., Bahrami, M., Fallah, M.S.: Local and Central Differential Privacy for Robustness and Privacy in Federated Learning. In: IEEE Int. Conf. on Big Data. pp. 2461–2469 (2020). `https://doi.org/10.1109/BigData50022.2020.9378333`
16. Nguyen, T.D., Rieger, P., Chen, H., et al.: FLAME: Taming backdoors in federated learning. In: 30th USENIX Security Symposium. pp. 1423–1440 (2021)
17. Nguyen, T.D., Rieger, P., Yalame, H., Möllering, H., Fereidooni, H., Marchal, S., Miettinen, M., Mirhoseini, A., Sadeghi, A.R., Schneider, T., et al.: Flguard: Secure and private federated learning. IACR Cryptol. ePrint Arch. **2021**, 25 (2021)

18. Nguyen, T.D., Nguyen, T., Nguyen, P.L., Pham, H.H., Doan, K., Wong, K.S.: Backdoor attacks and defenses in federated learning: Survey, challenges and future research directions (2023), https://arxiv.org/abs/2303.02213
19. Ozdayi, M.S., Dirik, A.E., Ozdemir, S.A.: Defending Against Backdoors in Federated Learning with Robust Learning Rate. In: AAAI Conf. on Artificial Intelligence. pp. 7465–7473 (2021). https://doi.org/10.1609/aaai.v35i9.16948
20. Shen, S., Tople, S., Saxena, P.: Auror: Defending against poisoning attacks in collaborative deep learning systems. In: Conf. on Computer Security Applications. pp. 508–519 (2016)
21. Sun, Z., Kairouz, P., Suresh, A.T., McMahan, H.B.: Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963 (2019)
22. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 30 (2017)
23. Walter, K., Sengupta, A., Raghunathan, A.: On the Limitations of Robust Aggregation for Federated Learning Backdoor Defenses. arXiv:2207.08433 (2022)
24. Walter, K., Nepal, S., Kanhere, S.: Exploiting layerwise feature representation similarity for backdoor defence in federated learning. In: European Symposium on Research in Computer Security. pp. 354–374. Springer (2024)
25. Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J.y., Lee, K., Papailiopoulos, D.: Attack of the tails: Yes, you really can backdoor federated learning. Advances in neural information processing systems **33**, 16070–16084 (2020)
26. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms (2017), http://arxiv.org/abs/1708.07747
27. Xie, C., Huang, K., Chen, P.Y., Li, B.: Dba: Distributed backdoor attacks against federated learning. In: International conference on learning representations (2019)
28. Zhang, Z., Cao, X., Jia, J., Gong, N.Z.: 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. pp. 2545–2555 (2022)

A   Appendix - Self-Attention Mechanism

Self-attention computes relationships between elements in a set, specifically client representation vectors r_i. For each target client i, our attention module computes a weighted average of other clients' vectors to approximate r_i.

Query, key, and value projection. We employ learnable projection matrices $W^Q, W^K, W^V \in \mathbb{R}^{d \times d_h}$ that map each d-dimensional representation into a lower-dimensional subspace of width d_h (typically $d_h = d/H$ for H attention heads). The server learns those matrices offline before the federated training begins. During this pre-training stage, the server minimises the reconstruction loss: $\mathcal{L}_{\text{rec}} = \frac{1}{m} \sum_{i=1}^{m} \|r_i - \hat{r}_i\|_2^2$, updating W^Q, W^K, W^V, and W^O with Adam ($\eta = 10^{-3}$, $(\beta_1, \beta_2) = (0.9, 0.999)$). This offline initialisation eliminates any reliance or assumptions about the proportion of benign clients during early training, while adding no extra computation during normal FL operation.

Attention weight computation. For head h and client i, attention scores between i and every other client $j \neq i$ are computed via scaled dot-products and normalized to obtain attention weights:

$$\alpha_{i \leftarrow j}^{(h)} = \frac{\exp\left(\frac{q_i^h \cdot k_j^h}{\sqrt{d_h}}\right)}{\sum_{t \neq i}^{m} \exp\left(\frac{q_i^h \cdot k_t^h}{\sqrt{d_h}}\right)} \tag{5}$$

Representation reconstruction. Using these weights, the reconstructed representation for client i is: $\hat{r}_i^{(h)} = \sum_{j \neq i} \alpha_{i \leftarrow j}^{(h)} v_j^h$. Concatenating head outputs, we obtain: $\hat{r}_i = \left(\|_{h=1}^{H} \hat{r}_i^{(h)}\right) W^O$. Where $W^O \in \mathbb{R}^{d \times d}$ is an output projection matrix. Typically, these matrices are learned via unsupervised reconstruction. Using multi-head attention (e.g., $H = 4$) provides marginal robustness improvements.

Robust anomaly score computation and client filtering. Once the original r_i and reconstructed $\hat{r}_i$ representations for each client i are computed, their discrepancy is quantified with a robust anomaly score. First, we compute the Euclidean reconstruction error: $e_i = \|r_i - \hat{r}_i\|_2$. Next, we calculate the median and median absolute deviation (MAD) to robustly detect anomalies. Clients with anomaly scores exceeding a threshold $\tau_{|z|}$ are marked as malicious.

Federated aggregation with filtering. Hard filtering discards the entire contribution of a suspicious client and may hurt learning when the benign population is small. Instead, we re-weight each update by soft attention weights:

$$\omega_i = \frac{\max\{0, \ \tau_{|z|} - |z_i|\}}{\sum_{j=1}^{m} \max\{0, \ \tau_{|z|} - |z_j|\}} \qquad (t > W) \tag{6}$$

so that a perfectly reconstructed client receives $\omega_i \approx 1/m$, while an extreme outlier ($|z_i| \geq \tau_{|z|}$) gets zero weight. The server then updates the global model with $w^{(t+1)} = \sum_{i=1}^{m} \omega_i w_i^{(t)}$. This attenuates suspicious gradients proportionally to their reconstruction error; when ω_i collapses to $\{0, 1/|C_{\text{trusted}}|\}$ it reduces to a hard filter.

B   Ablation studies on representation layer selection

Specifically, we evaluated (1) the final convolutional layer, (2) the penultimate fully-connected layer, and (3) global average pooling (GAP) layers in residual architectures. Table 7 summarizes the average anomaly detection accuracy (percentage of correctly identified malicious clients) for each representation choice, averaged across multiple rounds. The penultimate FC layer (CNNs) and the GAP layer (ResNets) deliver the best detection accuracy.

Table 7. Ablation results: anomaly-detection accuracy (%) for different representation layers.

Layer	CNN (MNIST)	CNN (CIFAR-10)	ResNet (CIFAR-10)
Final conv.	85.4	78.2	82.1
Penultimate FC	**98.7**	**95.3**	N/A
Global Avg. Pool	N/A	N/A	**96.2**