ViCAM-DFL: Visual Explanation-Driven Defenses against Model Poisoning in Decentralized Federated Learning-Enabled CyberEdge Networks

Jingjing Zheng*, Yu Gao*, Kai Li*†¶, Bochun Wu‡, Wei Ni§, and Falko Dressler†
*CISTER Research Centre, Portugal.
Email: zheng@isep.ipp.pt & up202204061@edu.med.up.pt
†Telecommunication Networks (TKN), TU Berlin, Germany.
Email: kaili@ieee.org & dressler@ccs-labs.org.
‡Fudan University, Shanghai 200433, China.
Email: wubochun@fudan.edu.cn.
§CSIRO, Australia.
Email: wei.ni@csiro.au.
¶Corresponding author

***Abstract*—In recent years, model poisoning attacks have emerged as a threat to the resilience of decentralized federated learning (DFL), as they corrupt model updates and compromise the integrity of collaborative training. To defend DFL against emerging model poisoning attacks based on graph neural networks, this paper proposes a specialized defense framework, visual explanation class activation mapping for DFL (ViCAM-DFL). The ViCAM-DFL transforms the high-dimensional local model updates into low-dimensional, visually interpretable heat maps that reveal adversarial manipulations. These heat maps are further refined using an integrated auto-encoder, which amplifies subtle features to enhance separability and improve detection accuracy. Experimental evaluations based on non-i.i.d. *CIFAR-100* datasets demonstrate that our ViCAM-DFL achieves substantial improvements in detecting adversarial manipulations. The framework consistently delivers optimal results in terms of key evaluation metrics, including *Recall, Precision, Accuracy, F1 Score*, and *AUC* (all reaching 1.0), while maintaining a *False Positive Rate (FPR)* of 0.0, outperforming baseline methods. Furthermore, ViCAM-DFL exhibits strong robustness and generalizability across different deep learning architectures, e.g., *ResNet-50* and *REGNETY-800MF*, confirming its adaptability and effectiveness in diverse DFL settings.**

***Index Terms*—Decentralized federated learning, model poisoning, resilience, defense frameworks, and visual explanations.**

I. Introduction

CyberEdge networks represent a next-generation architecture that combines mobile edge computing (MEC) with machine learning to deliver high-bandwidth, low-latency connectivity tailored for immersive metaverse applications, e.g., augmented reality (AR), virtual reality (VR), and mixed reality (MR) [1]. Ensuring user privacy and efficient bandwidth usage is essential in CyberEdge networks, where edge devices support real-time interactions and generate sensitive data, such as biometrics, geolocation, and behavioral patterns, making them attractive targets for malicious actors [2], [3].

To protect data privacy subject to bandwidth limitations, decentralized federated learning (DFL) has emerged as a promising solution in CyberEdge networks, enabling privacy-preserving and communication-efficient model training across distributed devices without exposing raw user data [4]–[8]. Specifically, each user communicates and exchanges model updates directly with its neighboring nodes in a peer-to-peer fashion. During training, each user updates its local model using private data and shares model parameters with its neighbors. These updates are aggregated locally at the user to align models across the network. This decentralized approach enhances robustness, trust, and scalability, making it particularly well-suited for metaverse applications in CyberEdge networks [9].

While DFL effectively mitigates data privacy leakage by keeping user data local, its distributed architecture also introduces critical vulnerabilities, particularly to model poisoning attacks. An attacker-controlled malicious user can deliberately alter local model parameters and propagate these compromised updates to neighboring nodes, leading to model corruption and significantly undermining the resilience of DFL [10]. For instance, as illustrated in Fig. 1, malicious users embed adversarial features or inject noise into their local updates, enabling them to bypass conventional poisoning defenses and degrade the performance of adjacent models.

To address the model poisoning attacks on DFL for enhancing the resilience, existing distance-based and machine learning-based defense mechanisms, e.g., Euclidean distance [11], cosine similarity [12], support vector machine (SVM) [13] and K-means [14], [15], have been developed to filter out suspicious or unreliable local models before aggregation. However, these defense measures face several challenges. First, excessive removal of local model updates and the costly analysis of high-dimensional local model updates may lead to inefficiency and significant degradation in model quality. Second, attackers can eavesdrop on benign local models to craft malicious updates that closely mimic them, thereby evading

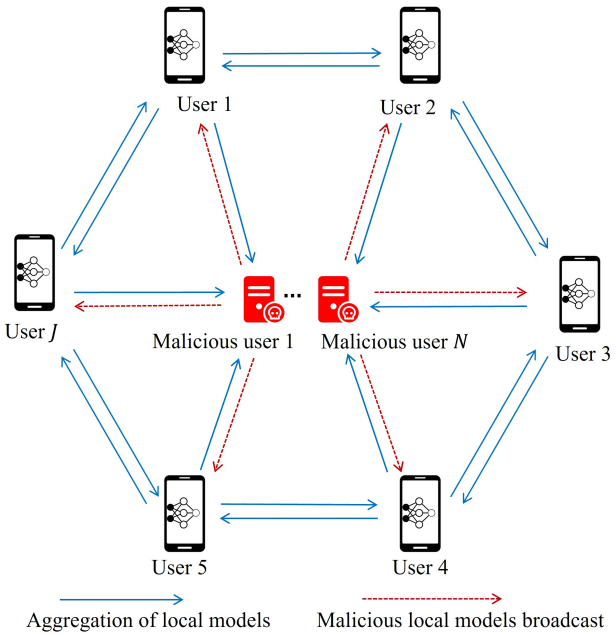

Fig. 1: Model poisoning attacks on DFL-enabled CyberEdge networks, where J benign users and N malicious users share locally trained model updates directly with their neighbors.

current defense mechanisms like Krum [16], Trimmed-mean [11], and Median [11]. Lastly, machine learning-based defense mechanisms are highly sensitive to parameters and suboptimal tuned parameters may lead to poor performance. Especially, the latest graph auto-encoder (GAE)-based model poisoning attacks [10] have been shown to bypass the existing distance-based defense mechanisms.

In this work, we propose a new visual explanation-based defense model, visual explanation class activation mapping for DFL (ViCAM-DFL), where ViCAM [17] is developed to create a heat map for every local model update of neighbors before aggregation. Due to the potential errors of ViCAM-assisted malicious user identification, an auto-encoder is smoothly incorporated in ViCAM-DFL to highlight the hidden features of the heat maps by remapping them, which improves the distinguishability of the heat maps and the success rate of discerning abnormal heat maps and malicious local model updates. Our key contributions include:

- Our proposed ViCAM-DFL leverages an extended visualization of class activation maps and auto-encoder to effectively detect inconspicuous manipulations, improving the resilience of DFL in CyberEdge networks.
- The ViCAM-DFL ingeniously exploits class activation maps to assist in transforming the high dimensional, indistinct local model updates in DFL into low-dimensional, visually interpretable heat maps.
- To eliminate potential errors of ViCAM-assisted malicious user identification, an auto-encoder is seamlessly incorporated into ViCAM-DFL, refining the heat maps to highlight their latent features and enhance their distinguishability. This improvement aids in more effectively identifying anomalous heat maps and detecting malicious local model updates.

The paper is organized as follows: Section II provides an overview of defense strategies based on Euclidean distance and machine learning. In Section III, we explore the DFL model and the threat model. The ViCAM-DFL defense framework is proposed in Section IV. We present our performance evaluation in Section V. The paper is concluded in Section VI.

II. Related Work

Several methods have utilized *Euclidean distance* to identify poisoned local model updates in DFL. Notably, approaches such as Krum [16] and its extension *Multi-Krum* [16] assign scores to local model updates by summing their Euclidean distances from neighboring updates. *Multi-Krum* subsequently filters out updates with the highest scores, thereby excluding potential outliers. Alternatively, the *Trimmed-mean* [11] algorithm adopts a coordinate-wise aggregation strategy, which reduces sensitivity to anomalous contributions. Beyond Euclidean distance-based defense strategies, *machine learning* models have been employed to identify malicious behavior in DFL. The *AUROR* framework utilizes K-means clustering to group local model updates [14]. Updates residing in small clusters beyond a predefined distance threshold are flagged as malicious and excluded. Another technique, federated anomaly analytics enhanced distributed learning (*FAA-DL*), employs an unsupervised SVM with a tailored kernel and soft-margin configuration to delineate nonlinear decision boundaries, effectively separating benign and adversarial contributions [13]. Given the importance of detecting CyberEdge network attacks, [18] introduced an FL framework that integrates an isolation forest algorithm. This approach identifies and filters malicious local model updates pre-aggregation by noting that malicious models tend to reside closer to the root in the forest's leaf nodes, thereby simplifying their detection. Furthermore, deep reinforcement learning was utilized to dynamically fine-tune the threshold for identifying these malicious updates.

Despite these advances, both Euclidean distance and machine learning-based defenses face challenges. In deep neural networks, the model updates may involve millions or even billions of parameters, which creates issues due to the "curse of dimensionality." In high-dimensional spaces, Euclidean distances become less meaningful and may fail to distinguish between malicious and benign model updates. Meanwhile, machine learning-based detection methods often require meticulous hyperparameter tuning and precise threshold setting, which may be unreliable. Recent research [19] further suggested these approaches may not consistently perform well in anomaly detection scenarios.

To overcome these challenges, our proposed ViCAM-DFL avoids relying on Euclidean distance-based metrics. This approach leverages ViCAM to convert complex, high-dimensional model updates into visual and low-dimensional heat maps. To further enhance discriminability, we integrate an auto-encoder that refines the heat maps by emphasizing subtle

features indicative of poisoning. By combining interpretability with dimensionality reduction, ViCAM-DFL significantly improves the robustness and accuracy of malicious model update detection in DFL CyberEdge networks.

III. Model Poisoning Attacks in Decentralized Federated Learning

A. Decentralized Federated Learning

Considering a DFL system with a set of users $\mathcal{V}$. We let $|\mathcal{V}|$ denote the number of users in the system. The network topology of this DFL system is defined by an undirected and unweighted communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{E}$ denotes the set of edges between users, and self-loops are not allowed. Communication is only possible between two users if there is an edge connecting them. Each individual user, denoted as $j \in \mathcal{V}$, has its own private dataset D_j.

Typically, the training procedure of DFL can be formulated as an empirical risk minimization (ERM) problem. For each user $j \in \mathcal{V}$ with $|D_j|$ training samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{|D_j|}$, model parameters $\mathbf{w}_j \in \mathbb{R}^d$, and a loss function $f_j(\mathbf{x}_i, y_i; \mathbf{w}_j)$, ERM can be defined as

$$\min_{\mathbf{w}_j} f_j(\mathbf{w}_j) = \frac{1}{|D_j|} \sum_{k=1}^{|D_j|} f(x_k, y_k; \mathbf{w}_j). \tag{1}$$

The DFL aims to solve the following problem in a fully distributed manner without requiring assistance from a server:

$$\mathbf{w}_j^* = \arg\min_{\mathbf{w}_j} f_j(\mathbf{w}_j), \tag{2}$$

where $\mathbf{w}_j^*$ is the optimal model weight parameters of j.

Local model training: Each user performs local training to get an intermediate model $\mathbf{w}_j^{(t+\frac{1}{2})}$, i.e.,

$$\mathbf{w}_j^{(t+\frac{1}{2})} = \mathbf{w}_j^{(t)} - \eta \nabla f_j\big(\mathbf{w}_j^{(t)}\big). \tag{3}$$

Local models aggregation: Each user subsequently sends $\mathbf{w}_j^{(t+\frac{1}{2})}$ to its neighboring user $i \in \mathcal{N}_j$, where $\mathcal{N}_j$ is the set of neighbors of user j, excluding the user j itself.

$$\mathbf{w}_j^{(t+1)} = \frac{|D_j|}{|D|} \mathbf{w}_j^{(t+\frac{1}{2})} + \mathrm{AGG}\left\{ \frac{|D_i|}{|D|} \mathbf{w}_i^{(t+\frac{1}{2})} \right\}, \tag{4}$$

where $|D| = |D_j| + \sum_{i \in \mathcal{N}_j} |D_i|$ and $\mathrm{AGG}\{\cdot\}$ is the aggregation function. The above two steps are repeated for multiple rounds until the DFL converges.

B. Threat Model

As in prior study [20], an attacker controls a subset of malicious users, which may either inject poisoned data into their local training or deliberately alter their model updates before transmitting them to neighboring nodes. It is important to highlight that each malicious user can only affect its direct neighbors by sending manipulated updates. In particular, the threat model described in [10] represents a specific instance of the general model poisoning attack considered in our paper, where the malicious update is generated to minimize the model accuracy of neighbors. By exploiting the feature correlation between benign neighbors' models, the attacker in [10] introduces subtle perturbations to the local model updates, while the malicious model remains undetected given the existing Euclidean distance-based or similarity-based defenses.

IV. Proposed ViCAM-DFL Defense Framework

We propose ViCAM-DFL, a novel malicious user detection architecture for DFL, grounded in visual interpretability. Each user j replaces its own model parameters with those models received from neighboring users one by one. After that, the user j visualizes all neighbor's model updates as a heat map leveraging the ViCAM technique on a test image. Although ViCAM allows high-dimensional model updates to be transformed into visually interpretable heat maps, these visualizations remain unlabeled and indistinct to the user. To address this, we incorporate an unsupervised auto-encoder into ViCAM-DFL, which does not require labeled data. The auto-encoder is trained to learn the latent patterns of typical (benign) heat maps, enabling it to detect anomalies by identifying significant deviations from this norm. This integration enhances the discriminative power of the heat maps and strengthens the system's ability to recognize irregular patterns linked to malicious updates.

The ViCAM-DFL architecture comprises two integral components: a ViCAM-driven module responsible for heat map generation and an auto-encoder-based mechanism for anomaly detection, as depicted in Fig. 2. Initially, ViCAM is applied to local model updates alongside a test image to produce activation heat maps. These maps are subsequently encoded and reconstructed by the auto-encoder, with reconstruction discrepancies serving as indicators of anomalous behavior.

ViCAM-based processing module. Each user j selects a random image from the test dataset and forwards it through convolutional layers whose weights and biases have been replaced with the intermediate local model updates $\mathbf{w}_i^{(t+\frac{1}{2})}, i \in \mathcal{N}_j$. This process yields a set of feature maps M_i with K channels. These feature maps are then passed through one or more fully connected (FC) layers to produce classification outputs. To derive a class-specific activation map $L_i^{(c)} \in \mathbb{R}^{W \times H}$ for a target class c, ViCAM calculates the gradient of the pre-softmax classification score $y^{(c)}$ with respect to each spatial location (p, q) of the k-th channel in the feature maps M_i, i.e., $\frac{\partial y^{(c)}}{\partial M_i^{(k)}(p,q)}$. Here, $p \in [1, W], q \in [1, H]$, and $k \in [1, K]$ represent the spatial and channel indices of the feature map. The relative importance of each spatial position, denoted by the weight $\alpha_{i,k}^{(c)}(p, q)$, is determined using the rectified linear unit (ReLU) activation function to emphasize positively contributing gradients, i.e.,

$$\alpha_{i,k}^{(c)}(p, q) = \mathrm{ReLU}\left(\frac{\partial y^{(c)}}{\partial M_i^{(k)}(p, q)} \right), \forall i \in \mathcal{N}_j, \tag{5}$$

where $M_i^{(k)}(p, q)$ denotes the activation at position (p, q) in the k-th channel of the feature map $M_i^{(k)}$. The ViCAM assigns an importance score to each spatial location and

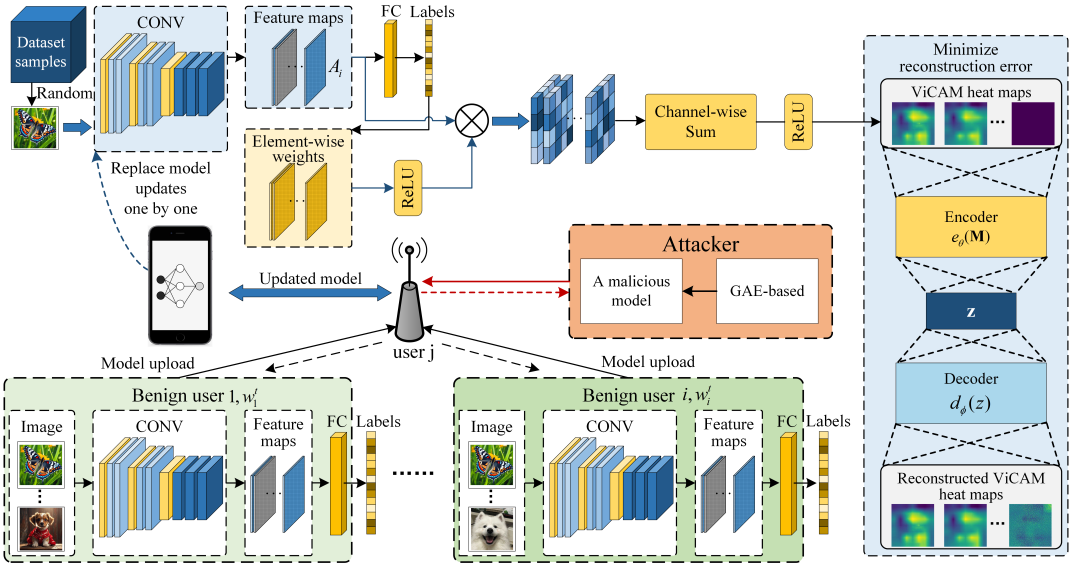

Fig. 2: Overview of the proposed ViCAM-DFL framework, where the user j samples an image (e.g., one labeled as "butterfly") from its test dataset. This image is processed by ViCAM to generate heat maps for the aggregated neighbors' model updates.

scales the corresponding activations accordingly. The weighted activations are aggregated across channels to generate the class-specific activation map, after which ReLU is applied to retain only positive contributions, as expressed by

$$M_i^{(c)}(p,q) = \text{ReLU}\left(\sum_{k=1}^{K} \alpha_{i,k}^{(c)}(p,q) \cdot M_i^{(k)}(p,q)\right). \quad (6)$$

Auto-encoder-based detection module. Each ViCAM-generated heat map $M_i^{(c)}(p,q)$, sized $W \times H$, is flattened into a $1 \times W \times H$ vector. These vectors from different neighbors are concatenated into a composite input matrix $\mathbf{M}$ for the encoder. Parameterized by θ, the encoder function $e_\theta(\cdot)$ projects $\mathbf{M}$ into a compact latent space, producing an embedding $z = e_\theta(\mathbf{M})$. This latent representation captures the essential features of the input heat maps. The decoder function $d_\phi(\cdot)$, with parameter ϕ reconstructs the original input heat maps from z, yielding $\mathbf{M}^{'} = d_\phi(z) = d_\phi\big(e_\theta(\mathbf{M})\big)$. The reconstructed heat maps are reshaped back to their original dimensions. Training is guided by minimizing the reconstruction loss, measured via the mean squared error (MSE) between input $\mathbf{M}$ and output $\mathbf{M}^{'}$:

$$\begin{aligned}\ell(\theta,\phi) &= \min_{\theta,\phi} \frac{1}{|\mathcal{N}_j|}\|\mathbf{M} - \mathbf{M}^{'}\|_2^2 \\ &= \min_{\theta,\phi} \frac{1}{|\mathcal{N}_j|} \sum_{i\in\mathcal{N}_j} \left\| M_i^{(c)} - d_\phi\Big(e_\theta\big(M_i^{(c)}\big)\Big)\right\|_2^2.\end{aligned} \quad (7)$$

After the auto-encoder training, the user j computes the reconstruction error E_i for each neighbor's heat map:

$$E_i = \frac{\sum_{p=1}^{W}\sum_{q=1}^{H} \left|M_i^{(c)}(p,q) - M_i^{'}(p,q)\right|}{W \times H}. \quad (8)$$

The average reconstruction error over j's all neighbors is:

$$\overline{E_j} = \frac{1}{|\mathcal{N}_j|} \sum_{i\in\mathcal{N}_j} E_i. \quad (9)$$

A dynamic threshold δ is computed as:

$$\delta = \overline{E_j} + \alpha \times \sqrt{\frac{\sum_{i\in\mathcal{N}_j}(E_i - \overline{E_j})^2}{|\mathcal{N}_j|}}, \quad (10)$$

where α is a tunable coefficient. The binary decision rule for each neighbor i at round t is defined as follows: if E_i is less than or equal to the threshold δ, the output $O_i^{(t)}$ is set to 1, indicating that the model is considered trusted (benign). Conversely, if E_i exceeds the threshold δ, $O_i^{(t)}$ is set to 0, indicating suspected malicious behavior. Malicious model updates are excluded from model aggregation. Therefore, the models aggregation for user j in Eq. (4), can be rewritten as

$$\mathbf{w}_j^{(t+1)} = \frac{|D_j|}{|D|}\mathbf{w}_j^{(t+\frac{1}{2})} + \text{AGG}\left\{O_i^{(t)} \cdot \frac{|D_i|}{|D|}\mathbf{w}_i^{(t+\frac{1}{2})}\right\}. \quad (11)$$

V. Performance Evaluation

We take the example that user j is connected to 24 peers, 3 of which are malicious participants. The ViCAM-DFL framework is configured with $T = 100$ communication rounds. Each user trains a local model for 25 epochs using the Adam optimizer with a batch size of 64 and a learning rate of 1e-4. For the auto-encoder, training is performed over 200 epochs using *Adam* with a learning rate of 1e-3, a hidden layer size of 128, and a weight decay (a hyperparameter of the penalty term of the local model loss function) of 1e-5. The tunable coefficient α is 1.50. All experiments are run on a single *GeForce RTX 4090 GPU* with 24 GB *GDDR6* memory.

To evaluate the effectiveness of defense mechanisms, we utilize the *ResNet-50* [21] and *REGNETY-800MF* [22] models trained on non-i.i.d. *CIFAR-100* data. In addition to the baseline methods *AUROR*, *Multi-Krum*, and *FAA-DL* (referenced in Section II), we also include two additional alternatives:

GCAMA [23], which uses *GradCAM* [24] with an auto-encoder to detect abnormal heat maps, and *LayerCAM-Krum*, which integrates LayerCAM-generated heat maps with the *Krum* [16] aggregation method for anomaly detection.

For quantifying detection performance, we use the following widely used metrics. i) *Recall*: the proportion of true malicious users correctly detected, relative to the total number of malicious users; ii) *Precision*: the fraction of users correctly identified as malicious among all users flagged as malicious by the defense strategies; iii) *false positive rate (FPR)*: the percentage of benign users that are mistakenly classified as malicious, relative to the total number of benign users; iv) *Accuracy*: the overall proportion of correctly classified users (both benign and malicious) among all users evaluated; v) *F1 Score*: the harmonic mean of precision and recall, which is computed using values from the confusion matrix; vi) *Area Under the ROC Curve (AUC)*: a scalar value ranging from 0 to 1 that reflects the capability of the defense to distinguish between benign and malicious users. A higher AUC value indicates better discriminatory power of the defense.

Fig. 3 plots the test accuracy of *ResNet-50* on the non-i.i.d. *CIFAR-100* dataset, demonstrating that our ViCAM-DFL achieves the highest test accuracy. Moreover, ViCAM-DFL can quickly converge (around the 10th round), as it involves more benign users in aggregation. This indicates that ViCAM-DFL can accurately filter malicious model updates, as can also be confirmed by the detection rates in Table I. Although *LayerCAM-Krum* can avoid malicious users being aggregated by the user j, it sacrifices accuracy and robustness, as it selects only one local model update as the updated model. The more divergent the local models, the more diverse the heat maps. The *LayerCAM-Krum* struggles to screen malicious models, which coincides with the precision of 0.637 for *LayerCAM-Krum* on non-i.i.d. *CIFAR-100*, as shown in Table I.

We replace the *ResNet-50* with the *REGNETY-800MF*. The trend of the test accuracy of the defenses with the communication rounds is consistent with the observation in Fig. 3, except for *LayerCAM-Krum*, as shown in Fig. 4. The reason is that the performance of *LayerCAM-Krum* may vary with the architecture of the neural network analyzed. It may not be as effective for models with complex architectures, such as attention-based models, where the relationships between features are more intricate. The *LayerCAM-Krum* mistakenly selects the malicious model as the model update in the 68th communication round, causing the test accuracy of the model to drop sharply. The detection rates of *ResNet-50* on non-i.i.d. *CIFAR-100* are given in Table I.

Compared to *GCAMA*, ViCAM-DFL is designed to be compatible with various network architectures, including both traditional DNNs and more complex model architectures such as *ResNets* or *REGNETY-800MF*. This flexibility enables ViCAM-DFL seamless integration with diverse model types without requiring structural modifications. In essence, the performance of ViCAM-DFL remains robust and largely invariant to the underlying network architecture, a trend clearly observed in Fig. 3 and Fig. 4. This architectural robustness is

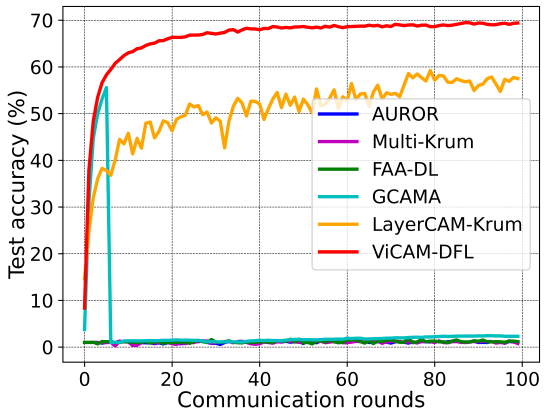

Fig. 3: The test accuracy of *ResNet-50* on non-i.i.d. *CIFAR-100*.

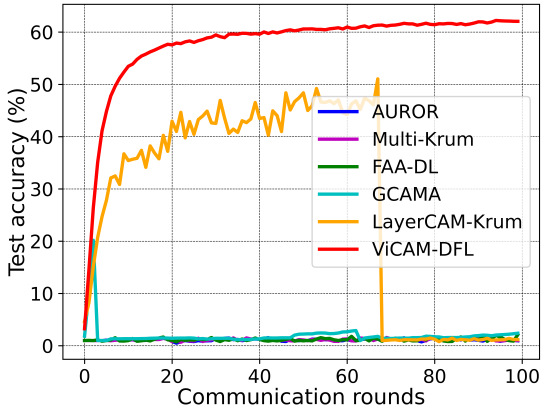

Fig. 4: The test accuracy of *REGNETY-800MF* on non-i.i.d. *CIFAR-100*.

further validated by the results in Table I, where ViCAM-DFL consistently outperforms baseline methods, achieving near-ideal scores across all evaluation metrics, i.e., *Recall*: 1.0, *Precision*: 1.0, *FPR*: 0.0, *Accuracy*: 1.0, *F1 score*: 1.0, and *AUC*: 1.0. Specifically, we assume that the server and benign users operate on the same datasets in our proposed system, while the attackers have no direct knowledge of the underlying data. The server detects potentially malicious models by comparing the heat maps generated by the server for local models using the shared dataset; any significant divergence can be used to identify a malicious model with highly accurate identification. In contrast, baseline methods based on distance or similarity measures exhibit degraded detection performance when malicious models are highly correlated with benign ones.

VI. Conclusion

In this paper, we proposed a novel visual explanation-driven defense strategy, ViCAM-DFL, to enhance the resilience of DFL against model poisoning attacks. The proposed ViCAM-

TABLE I: Detection rates of *ResNet-50* and *REGNETY-800MF* on non-i.i.d. *CIFAR-100*.

non-i.i.d. *CIFAR-100*	*ResNet-50*						*REGNETY-800MF*					
Methods	*Recall*	*Precision*	*FPR*	*Accuracy*	*F1 score*	*AUC*	*Recall*	*Precision*	*FPR*	*Accuracy*	*F1 score*	*AUC*
AUROR	0.020	0.013	0.181	0.718	0.016	0.419	0.013	0.008	0.158	0.738	0.01	0.427
Multi-Krum	0.077	0.077	0.132	0.769	0.077	0.472	0.1	0.1	0.129	0.775	0.1	0.486
FAA-DL	0.703	0.128	0.680	0.368	0.215	0.512	0.727	0.123	0.730	0.327	0.210	0.498
GCAMA	1.0	0.95	0.010	0.992	0.971	0.999	0.828	0.828	0.02	0.974	0.820	0.917
LayerCAM-Krum	0.337	0.337	0.095	0.834	0.337	0.621	0.95	0.95	0.007	0.987	0.95	0.971
ViCAM-DFL	**1.0**	**1.0**	**0.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**0.0**	**1.0**	**1.0**	**1.0**

DFL leverages an auto-encoder-assisted defense mechanism to detect adversarial manipulations in local model updates. The defense operates by applying a representative test image to generate ViCAM-based heat maps from local updates, which are refined using an extended auto-encoder to enhance the visibility of subtle, embedded features. Experimental results on the non-i.i.d. *CIFAR-100* dataset demonstrate that ViCAM-DFL achieves exceptional detection performance, attaining near-ideal detection rates across all evaluation metrics (*Recall*: 1.0, *Precision*: 1.0, *FPR*: 0.0, *Accuracy*: 1.0, *F1 Score*: 1.0, and *AUC*: 1.0), and significantly outperforming baseline methods. Furthermore, ViCAM-DFL consistently delivers optimal results across various deep learning architectures, confirming its robustness and adaptability.

ACKNOWLEDGMENT

This Paper was supported by the CISTER Research Unit (UIDP/UIDB/04234/2020), financed by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), by project ADANET (PTDC/EEICOM/3362/2021), and by project Aero.Next Portugal (ref. C645727867-00000066), funded by the EU/Next Generation, within call n.o 02/C05-i01/2022 of the Recovery and Resilience Plan (RRP), and under the project Intelligent Systems Associate Laboratory - LASI (LA/P/0104/2020).

REFERENCES

[1] K. Li, Y. Cui, W. Li, T. Lv, X. Yuan, S. Li, W. Ni, M. Simsek, and F. Dressler, "When Internet of Things meets metaverse: Convergence of physical and cyber worlds," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4148–4173, 2023.

[2] K. Li, Z. Zhang, A. Pourkabirian, W. Ni, F. Dressler, and O. B. Akan, "Towards resilient federated learning in cyberedge networks: Recent advances and future trends," *arXiv preprint arXiv:2504.01240*, 2025.

[3] K. Li, C. Li, X. Yuan, S. Li, S. Zou, S. S. Ahmed, W. Ni, D. Niyato, A. Jamalipour, F. Dressler, and O. B. Akan, "Zero-trust foundation models: A new paradigm for secure and collaborative artificial intelligence for internet of things," *IEEE Internet of Things Journal*, pp. 1–1, 2025.

[4] E. T. Martínez Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2983–3013, 2023.

[5] K. Li, J. Zheng, W. Ni, H. Huang, P. Liò, F. Dressler, and O. B. Akan, "Biasing federated learning with a new adversarial graph attention network," *IEEE Transactions on Mobile Computing*, vol. 24, no. 3, pp. 2407–2421, 2025.

[6] K. Li, X. Yuan, J. Zheng, W. Ni, F. Dressler, and A. Jamalipour, "Leverage variational graph representation for model poisoning on federated learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 1, pp. 116–128, 2025.

[7] G. Chen, K. Li, A. M. Abdelmoniem, and L. You, "Exploring representational similarity analysis to protect federated learning from data poisoning," in *the ACM Web Conference*, p. 525–528, 2024.

[8] J. Zheng, K. Li, X. Yuan, W. Ni, E. Tovar, and J. Crowcroft, "Exploring visual explanations for defending federated learning against poisoning attacks," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, p. 1596–1598, 2024.

[9] L. Yuan, Z. Wang, L. Sun, P. S. Yu, and C. G. Brinton, "Decentralized federated learning: A survey and perspective," *IEEE Internet of Things Journal*, vol. 11, no. 21, pp. 34617–34638, 2024.

[10] K. Li, J. Zheng, X. Yuan, W. Ni, O. B. Akan, and H. V. Poor, "Data-agnostic model poisoning against federated learning: A graph autoencoder approach," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 3465–3480, 2024.

[11] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 5650–5659, PMLR, 10–15 Jul 2018.

[12] 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, February 21-25, 2021*.

[13] S. Shi, C. Hu, D. Wang, Y. Zhu, and Z. Han, "Federated anomaly analytics for local model poisoning attack," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 2, pp. 596–610, 2022.

[14] S. Shen, S. Tople, and P. Saxena, "Auror: defending against poisoning attacks in collaborative deep learning systems," in *the 32nd Annual Conference on Computer Security Applications*, p. 508–519, 2016.

[15] X. Feng, W. Cheng, C. Cao, L. Wang, and V. S. Sheng, "DPFLA: Defending private federated learning against poisoning attacks," *IEEE Transactions on Services Computing*, pp. 1–12, 2024.

[16] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] P.-T. Jiang, C.-B. Zhang, Q. Hou, M.-M. Cheng, and Y. Wei, "LayerCAM: Exploring hierarchical class activation maps for localization," *IEEE Transactions on Image Processing*, vol. 30, pp. 5875–5888, 2021.

[18] Z. Chen, A. Fu, Y. Zhang, Z. Liu, F. Zeng, and R. H. Deng, "Secure collaborative deep learning against GAN attacks in the Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5839–5849, 2021.

[19] S. Han, X. Hu, H. Huang, M. Jiang, and Y. Zhao, "ADBench: Anomaly detection benchmark," in *Advances in Neural Information Processing Systems, November 28 - December 9, 2022*.

[20] M. Fang, Z. Zhang, Hairi, P. Khanduri, J. Liu, S. Lu, Y. Liu, and N. Gong, "Byzantine-robust decentralized federated learning," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, p. 2874–2888, 2024.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[22] I. Radosavovic, R. P. Kosaraju, R. B. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 13-19, 2020*, pp. 10425–10433.

[23] J. Zheng, K. Li, X. Yuan, W. Ni, and E. Tovar, "Detecting poisoning attacks on federated learning using gradient-weighted class activation mapping," in *the ACM on Web Conference*, p. 714–717, 2024.

[24] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-Cam: Visual explanations from deep networks via gradient-based localization," in *IEEE International Conference on Computer Vision*, pp. 618–626, 2017.