

FreqFed: A Frequency Analysis-Based Approach for Mitigating Poisoning Attacks in Federated Learning

Hossein Fereidooni

Technical University of Darmstadt¹
hossein.fereidooni@trust.tu-darmstadt.de

Alessandro Pegoraro

Technical University of Darmstadt
alessandro.pegoraro@trust.tu-darmstadt.de

Phillip Rieger

Technical University of Darmstadt
phillip.rieger@trust.tu-darmstadt.de

Alexandra Dmitrienko

University of Würzburg
alexandra.dmitrienko@uni-wuerzburg.de

Ahmad-Reza Sadeghi

Technical University of Darmstadt
ahmad.sadeghi@trust.tu-darmstadt.de

Abstract—Federated learning (FL) is a collaborative learning paradigm allowing multiple clients to jointly train a model without sharing their training data. However, FL is susceptible to poisoning attacks, in which the adversary injects manipulated model updates into the federated model aggregation process to corrupt or destroy predictions (untargeted poisoning) or implant hidden functionalities (targeted poisoning or backdoors). Existing defenses against poisoning attacks in FL have several limitations, such as relying on specific assumptions about attack types and strategies or data distributions or not sufficiently robust against advanced injection techniques and strategies and simultaneously maintaining the utility of the aggregated model.

To address the deficiencies of existing defenses, we take a generic and completely different approach to detect poisoning (targeted and untargeted) attacks. We present *FreqFed*, a novel aggregation mechanism that transforms the model updates (i.e., weights) into the frequency domain, where we can identify the core frequency components that inherit sufficient information about weights. This allows us to effectively filter out malicious updates during local training on the clients, regardless of attack types, strategies, and clients' data distributions. We extensively evaluate the efficiency and effectiveness of *FreqFed* in different application domains, including image classification, word prediction, IoT intrusion detection, and speech recognition. We demonstrate that *FreqFed* can mitigate poisoning attacks effectively with a negligible impact on the utility of the aggregated model.

I. INTRODUCTION

Federated Learning (FL) is a distributed machine learning paradigm that enables collaboration in training a global model by multiple clients without sharing their own local data. FL is based on the concept of federated optimization, where each client performs local optimization on its own data and exchanges model parameters with a central server, which aggregates the information to update the global model. The aggregated model is then returned to each client for the next training iteration. By design, the global server is uninformed about the local training process for individual clients. This,

however, makes FL vulnerable to model and data poisoning attacks launched by malicious clients. Recently, researchers have shown various poisoning attacks on FL, in which the adversary injects manipulated model updates into the federated model aggregation process to destroy or corrupt the resulting predictions (a.k.a. untargeted poisoning) [5], [15], [46], or implants hidden functionalities (a.k.a. targeted poisoning or backdoors) [47], [39], [4], [52], [57].

Current defenses against poisoning attacks (targeted and untargeted) are typically entangled with inspecting or directly computing with models' weights, such as using output predictions [10], [1], intermediary states (i.e., logits) [44], or different norms (e.g., L_2 – norm or cosine distance) among local models or between local models and the global model [6], [34], [9], [38], [16].

These approaches, however, lead to the following significant limitations: Firstly, the adversary can manipulate the model's weights to influence defense-related metrics computed over weights, thereby evading anomaly detection algorithms [4], [52]. Secondly, the defense mechanisms that apply Differential Privacy (DP) directly operate on weights to add noise and perform clipping, which decreases the model's overall utility [4], [31], [35]. Furthermore, these defenses rely on certain assumptions regarding client data distributions (i.e., iid² or non-iid) [38], [47], [34], [16], [61], [6], [9], [27] and attack types and strategies [1], [6], [35], [24], [16], which can limit their effectiveness resulting in a less generic adversary model. These assumptions can lead to deterioration in model accuracy when not met [61], [6], [9]. In particular, if defenses assume that all clients' data follow a similar distribution, they struggle to distinguish between malicious and benign clients with data from different distributions.

Mainly, defenses against backdoor attacks [38], [4], [57], [15], [44], [52] can be bypassed by adaptive attacks and strategies like multiple backdoors [4], distributed backdoors [57], and advanced techniques (e.g., Constrain-and-Scale [4] and Projected Gradient Descent (PGD) [51], [52]).

¹The author worked on this project while being affiliated with TU Darmstadt but is now at KOBIL GmbH.

²iid: independent and identically distributed

These limitations highlight the need for more generic, robust, and effective solutions against poisoning attacks. To this end, as discussed above, we investigate a completely different approach to designing a defense against poison attacks that untangles the defense from direct inspection and operation on weights or specific assumptions about the adversary and data distributions. Recently, Kumari et al. [24] introduced an alternate representation of the client updates, a probabilistic measure over the weights. Based on this probabilistic measure, they designed a detection mechanism that filters malicious updates, more precisely, backdoors. While their proposal is a significant advancement in untangling the detection mechanism from the data distributions and attack strategies, it is currently ineffective against untargeted and multiple backdoor attacks. Our work improves their approach [24] significantly and effectively mitigates targeted and untargeted attacks.

Goals and Contributions. We aim to tackle the limitations of current defenses by presenting the design and implementation of *FreqFed*, a resilient aggregation framework for FL that efficiently eliminates the impact of both targeted and untargeted poisoning attacks while retaining the benign performance of the aggregated model. Our method transforms the weights into the frequency domain, where they are interpreted as signals. The frequency components encode sufficient information and are robust against manipulation to discern benign and malicious model weights. Transforming weights into another domain facilitates decision-making in detecting attacks. To the best of our knowledge, this is the first work that applies the frequency analysis method on model weights to design a robust aggregation framework for FL against poisoning attacks. In particular, our contributions are as follows:

- We present *FreqFed*, a defense against poisoning attacks in FL that accurately and effectively mitigates targeted and untargeted attacks without significantly affecting the performance of the aggregated model. Our defense does not directly administer client updates (model weights) and operates under a general adversary model assumption (as described in Sections III and IV).
- We employ a frequency analysis method, Discrete Cosine Transform (DCT), to transform local model weights into the DCT domain. The intent behind this methodology is to discern the impact of poisoning attacks on the distribution of weights and their corresponding energies in a Neural Network (NN) model. Our approach relies on two central observations. Firstly, the predominance of energy within the model weights is situated within the low-frequency DCT components [54], [59]. Secondly, throughout the training process, NN models prioritize low frequencies and progress from low to high frequencies when approximating target functions [42], [60]. As a result of these observations, we were inspired to undertake a more in-depth examination of the low-frequency DCT spectrum. This enabled us to incorporate the low-

frequency components into our automated clustering approach (HDBSCAN), thereby allowing us to identify and eliminate potentially poisoned model updates (see Section IV-B).

- We perform comprehensive experiments using various datasets and models such as Deep Neural Networks (DNNs) and Graph Neural Networks (GNNs) in different application domains like image and graph classification (IC, GC), word prediction (WP), network intrusion detection (NIDS) and speaker verification (SV). Our evaluation shows *FreqFed* is independent of data distributions (whether iid or non-iid.), attack types (targeted or untargeted), attack strategies (e.g., adaptive), and poison injection techniques while preserving the overall performance of the global model (see Section V).

II. BACKGROUND AND PRELIMINARIES

In the following, we provide an overview of the background and preliminaries that set the foundation for the research presented in this paper.

A. Federated Learning

Federated Learning (FL) enables multiple clients to collectively train a global model without sharing their data. During each training round t , client i ($i \in 1, \dots, K$) trains its local model using its private data d_i and the parameters of the previous global model G_t as the starting point. After training, each client i sends the updated parameters of its local model W_i^t to the server S . The server then aggregates the received model updates using a specified aggregation rule to obtain the updated global model G_{t+1} , which is used as the global model for the next round $t+1$ and distributed to the clients [30]. In this paper, we will make use of FedAVG [30] as the aggregation rule:

$$G_{t+1} = \sum_{i=1}^K \frac{n_i}{n} W_i^t, \quad (1)$$

where n_i is the number of samples for client i , and n is the number of samples for all clients.

B. Poisoning attacks in FL

Poisoning attacks in FL can be broadly divided into two categories: untargeted and targeted. Untargeted attacks are designed to impair the performance of the aggregated model and hinder its generalization capabilities, as highlighted in [5], [15], [46]. In contrast, targeted attacks aim to implant hidden functionalities (backdoors) into the model [4], [39], [51], [57]. Such attacks allow the adversary to control the model's behavior stealthily. These attacks are particularly dangerous as they can go unnoticed for an extended period and lead to serious security breaches.

C. Discrete Cosine Transform

In the signal processing domain, Discrete Cosine Transform (DCT) [36] is leveraged to decompose a signal into frequency components, revealing the dynamics that make up the signal and transitions within it [56]. The DCT represents a finite sequence of data points as the sum of sinusoids with different frequencies and amplitudes. The DCT, particularly the two-dimensional DCT (2d DCT), is frequently utilized in signal processing and data compression because it has a strong *energy compaction* property and can pack input data into as few coefficients as possible.

Mathematically, DCT transformations are invertible functions that map an input sequence of N real numbers to the coefficients of N orthogonal cosine basis functions of increasing frequencies. The DCT components are listed in ascending order of significance. The first coefficient is proportional to the sequence average and represents the sum of the input sequence normalized by the square length. The lower-order coefficients represent lower signal frequencies correlating to the sequence's patterns. The following equation will give the 2d DCT of a signal x (e.g., N by M matrix) with frequencies of k and l [12], [55].

$$X(k, l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} c_1 c_2 x(m, n) \cos\left(\frac{k\pi}{2M}(2m+1)\right) \cos\left(\frac{l\pi}{2N}(2n+1)\right) \quad (2)$$

Where $c_1, c_2, k,$ and l are:

$$\begin{cases} c_1 = \sqrt{\frac{2}{MN}} & \text{for } k = 0, & c_1 = 1 & \text{for } k = 1, 2, \dots, M-1 \\ c_2 = \sqrt{\frac{2}{MN}} & \text{for } l = 0, & c_2 = 1 & \text{for } l = 1, 2, \dots, N-1 \end{cases}$$

D. HDBSCAN Clustering³

HDBSCAN [29] is an advanced density-based clustering technique that utilizes the density-reachability principle for identifying data clusters. It assesses the proximity of data points and the density of their distribution to determine group membership, allowing for the analysis of complex and highly variable data. One of the critical advantages of HDBSCAN is its ability to identify clusters of different sizes and shapes. Traditional clustering algorithms, including K-Means, have limitations in recognizing clusters that vary in size and require a prior determination of the desired number of clusters. HDBSCAN can address these limitations and is more robust to the shape and size of the clusters. Another advantage of HDBSCAN is its ability to identify clusters that contain noise or outliers. Traditional clustering algorithms may struggle to identify clusters in the presence of noise or outliers, as they can disrupt the formation of clusters. HDBSCAN can identify clusters even in the presence of noise or outliers, making it a valuable tool for data preprocessing and clustering.

³Hierarchical Density-Based Spatial Clustering of Applications with Noise

III. ADVERSARY MODEL

The adversary's objective is either to render the global model useless and eventually lead to denial-of-service (untargeted attacks) or to insert backdoors into the global model to cause misclassification based on a set of attacker-chosen inputs with high confidence (targeted attack). The adversary also aims to maintain the high accuracy of the aggregated model on both the main task and the adversary-chosen subtask. We do not assume specific data distributions (i.e., iid/non-iid) for the attacker and benign clients participating in FL training.

Attacker's Capabilities. The adversary can:

- Fully control up to $k_A < K/2$ compromised clients, including the entire local training data of these clients, as well as the local training operation and the hyperparameters (i.e., learning rate, number of training epochs, etc.). This assumption is aligned with related work (e.g., [1], [44], [47], [43]).
- Manipulate the weights of the resulting local model before submitting it to the global server for aggregation. Still, the adversary has no control over any processes executed at the aggregator or the honest clients. However, the attacker can know the global server's aggregating operations perfectly.
- Maliciously craft model updates by adding regularization terms to the loss function to evade the global server's anomaly detector's detection scope and make poisoned models as indistinguishable as possible from benign ones. Thus, the adversary ensures that any comparable detection metric between the poisoned and the benign model (e.g., L_2 - norm, cosine angular distance, etc.) is less than some threshold τ . The adversary can calculate this threshold while training the local model on benign data to always remain among the benign clients.
- Change its local training from round to round and always decide to behave normally or maliciously in a specific training iteration. We make no particular assumptions about the adversary's behavior.
- Conduct adaptive attacks by manually tweaking attack parameters (i.e., poison model rate, poison data rate, training loss, etc.) to exploit weak points of the deployed defense mechanism. The adversary can also follow any injection strategies using state-of-the-art injection techniques.

IV. DESIGN

This section presents the design and implementation of *FreqFed*. First, we discuss the design challenges of *FreqFed*, including the need to be agnostic to the types and strategies of attacks and the underlying data distributions. Next, we outline the high-level idea and intuition of our approach. Finally, we provide a detailed description of *FreqFed*, including its system overview and the components of the defense mechanism.

A. Design Challenges and Ideas

The mitigation of poisoning attacks in FL presents several challenges: (i) there is a need to effectively filter out malicious updates regardless of the type of attack (i.e., targeted or untargeted [47], [15], [28], [4], [57], [51] attacks, optimized [38], [15], [44], [46] or non-optimized attacks [47], [57], [4], [52]), (ii) the solution must be resilient against adaptive attack strategies [15], [44], robust against diverse backdoor injection techniques (i.e., single/multiple [4], [51] or distributed backdoors [57]), and independent of underlying data distributions (e.g., iid or non-iid), and (iii) it is crucial that the proposed solution does not sacrifice the utility of the global model in the process of removing malicious updates.

To address these challenges *simultaneously*, determining a substitute representation for the model weights is of paramount importance. The alternative must encode enough information about the weights and be resilient to tampering in various poisoning attacks. Our work accomplished this using a frequency analysis method called the Discrete Cosine Transform (DCT). Our intuition is that frequency analysis of local model updates in the frequency domain would allow us to identify patterns unique to malicious updates. We posit that malicious updates intended to introduce a backdoor into the global model or impact the overall performance of the global model are characterized by differences in the low-frequency components compared to benign updates.

To clarify our idea, we will discuss our intuition as follows: In a Neural Network (NN) model, each weight represents the strength of the connection between two neurons. The weight distribution and associated energies undergo a dynamic evolution during training to bridge the gap between the model’s predictions and the true targets within the training data. When a NN model is subjected to poisoning attacks, for instance, backdoor attacks, inserting a backdoor implies that some training data have been manipulated consistently to associate a certain input pattern (i.e., the backdoor trigger) with a specific output. This forces the model to learn a new, artificial correlation that would not be present in the benign data. This artificial correlation represents an additional structure or pattern in the model’s weights imposed by the backdoor trigger. Depending on how the backdoor trigger is designed and implemented, this pattern could be either subtle or quite distinct, but either way, it represents a deviation from the patterns the model would have learned from the normal data.

Since the DCT of the weights is a representation of how the energy of the weights is distributed across different frequencies, introducing backdoor changes this distribution, causing a shift in the energy towards certain frequencies. We made two central observations: i) most of the energy in model weights lies in low-frequency DCT components [54], [59], and ii) DNNs prioritize low frequencies and progress from low to high frequencies when approximating target functions

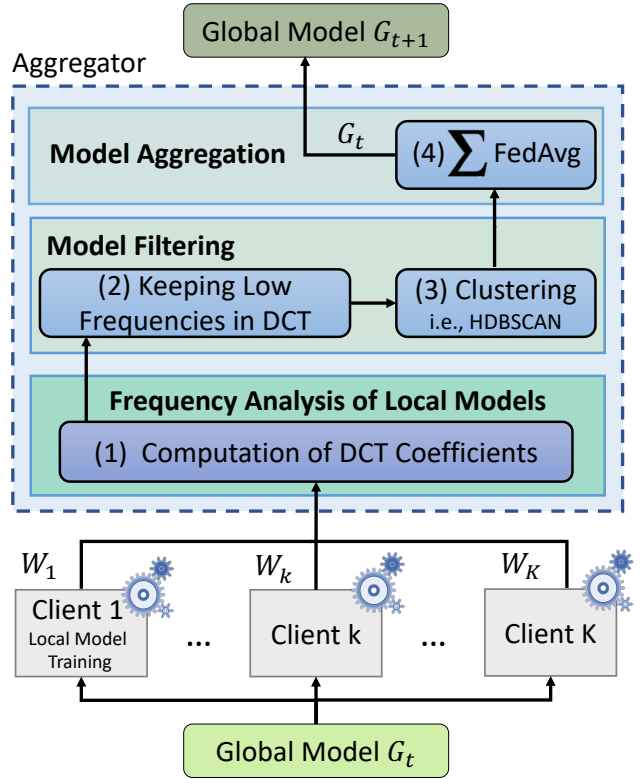


Fig. 1: System Overview of *FreqFed*

during training [42], [60]. These observations encouraged us to focus more on the low-frequency DCT spectrum and scrutinize if backdoors cause an energy shift in the low-frequency components of the DCT.

B. High-level Overview

The high-level overview of our framework *FreqFed* is shown in Figure 1. *FreqFed* comprises three critical components for i) frequency analysis of local model updates, ii) model filtering, and iii) model aggregation. In the following, we give a brief explanation of the functionality of each component.

Frequency analysis of local model updates obtains the model updates from each client participating in the federated learning process and transforms them subsequently into the frequency domain using Discrete Cosine Transform in Step 1. This process is used to identify the dominant frequencies in the updates, which can subsequently be used for model filtering.

Model filtering is in charge of processing the matrices of DCT coefficients and extracting the low-frequency components, which are then stored in a vector. The low-frequency component vectors from all clients are subsequently passed to a clustering algorithm, which groups them into clusters based on cosine distance and selects the cluster with the most vectors in Steps 2 and 3. This process is used for identifying the updates that are most representative of the majority of

clients, which can be subsequently used for model aggregation.

Model aggregation aggregates the model updates corresponding to the vectors in the chosen cluster, utilizing the Federated Averaging algorithm (see Section II-A) to update the global model in Step 4. This process allows for integrating updates from multiple clients into a joint global model.

C. FreqFed Design Details

We describe the algorithm used to implement our proposed framework in the following. We provide a detailed explanation of modules in the algorithm, including their purpose and any relevant implementation details. By providing this information, we aim to give the reader a clear understanding of how the framework operates and how the algorithms contribute to its overall functionality. Algorithm 1 outlines the entire flow of *FreqFed*. It takes as input the number of clients participating in the FL process K , the randomly initialized global model G_1 , and the number of training iterations T . The output G_{T+1} is the updated global model after T iterations. For each training iteration t in the range $[1, T]$, the algorithm cycles over each client i in the range $[1, K]$. The client’s model update, W_i , is obtained by calling the `ClientUpdate` procedure with the previous global model, G_t , and the current client i as input. The model update is then transformed into a matrix of coefficients, V_i , using the Discrete Cosine Transform (DCT) with the `DCT` procedure. The low-frequency components, F_i , of the matrix, V_i , are then extracted using the `Filtering` procedure.

Once all clients have submitted their model updates, the `Clustering` procedure is invoked with the low-frequency components extracted from all the clients’ model updates, F_1, \dots, F_k , as input. The `Clustering` procedure returns a list of indices, (b_1, \dots, b_L) , corresponding to the elements in the largest cluster, i.e., the list of indices of accepted models. The global model is then updated by aggregating the accepted models, W_{b_l} , for l in the range $[1, L]$, and dividing by the number of accepted models, L . This process is repeated for the remaining training iterations until the final global model, G_{T+1} , is obtained.

The process for updating the local models through training is described in lines 8-12. In local training, each client i trains its model based on its local data, using the initial random global model as a starting point. The client’s model w is then updated using the local gradients computed from its data, and these updates are communicated to the global server.

The procedure for filtering DCT coefficients is described in lines 13-18. After computing the DCT representation of the model weights, the low-frequency components are extracted as they are considered the most crucial frequency coefficients. The algorithm starts with a square matrix of DCT components V and creates a vector F to store the low-frequency components. It then loops through the elements of V and adds the low-frequency features (i.e., those with indices $i + j \leq \lfloor |V|/2 \rfloor$) to F . The resulting vector F contains the low-frequency components of the model update.

Algorithm 1 *FreqFed* (K, G_1, T)

Input: K clients, G_1 initial global model, T number of rounds

Output: G_{T+1} updated global model

Server executes:

- 1: **for** each training iteration t in $[1, T]$ **do**
- 2: **for** each client i in $[1, K]$ **do**
- 3: $W_i^t \leftarrow \text{ClientUpdate}(G_t, i)$
- 4: $V_i^t \leftarrow \text{DCT}(W_i^t)$
- 5: $F_i^t \leftarrow \text{Filtering}(V_i^t)$
- 6: $(b_1, \dots, b_L) \leftarrow \text{Clustering}(F_1^t, \dots, F_k^t)$
- 7: $G_{t+1} \leftarrow \sum_{l=1}^L W_{b_l}^t / L$
- return** G_{T+1}

- 8: **function** `CLIENTUPDATE`(w, i) $\triangleright w = G_1$ in first round
- 9: $\beta \leftarrow (\text{Split training data into batches of size } B)$
- 10: **for** each local epoch e in $[1, E_i]$ **do**
- 11: **for** each batch $b \in \beta$ **do**
- 12: $w \leftarrow w - \eta_i \nabla \ell(w; b)$
- return** w **to server**

- 13: **function** `FILTERING`(V)
- 14: $F \leftarrow \emptyset$
- 15: **for** i in $[0, \lfloor |V|/2 \rfloor]$ **do**
- 16: **for** j in $[0, \lfloor |V|/2 \rfloor]$ **do**
- 17: **if** $i + j \leq \lfloor |V|/2 \rfloor$ **then**
- 18: $F \cup V_{ij}$
- return** F

- 19: **function** `CLUSTERING`(F_1, \dots, F_k)
- 20: $\text{distances_matrix} \leftarrow \text{Initialized } \emptyset$
- 21: **for** each i in $[1, K]$ **do**
- 22: **for** each j in $[1, K]$ **do**
- 23: $\text{distances_matrix}_{ij} \leftarrow 1 - \text{CosineSim}(F_i, F_j)$
- 24: $\text{distances_matrix}_{ji} \leftarrow \text{distances_matrix}_{ij}$
- 25: $\text{cluster_ids} \leftarrow \text{HDBScan}(\text{distances_matrix})$
- 26: $\text{max_cluster} \leftarrow \arg_a \max \{ |a \in \text{cluster_ids}| \}$
- 27: $B \leftarrow \emptyset$
- 28: **for** i in $[1, K]$ **do**
- 29: **if** $\text{cluster_ids}_i = \text{max_cluster}$ **then**
- 30: $B \cup i$
- return** B $\triangleright B: b_1, \dots, b_L$

The `Clustering` procedure aims to automatically identify and remove malicious updates by clustering the low-frequency components of model updates based on their cosine distance. As shown in lines 19-30, it takes as input the vectors of low-frequency components, F_1, \dots, F_k , of the model updates. It returns a list of indices b_1, \dots, b_L corresponding to the accepted models. The number of accepted models L is the size of the most significant cluster identified by the algorithm. The `Clustering` module initializes a matrix $K \times K$ of cosine distances distances_matrix with all zero values to cluster the

model updates. Then, it computes the cosine distance between every pair of low-frequency component vectors as 1 minus the Cosine Similarity of F_i and F_j . All distances are stored in the matrix *distances_matrix* at indices ij and ji . Next, the Clustering procedure utilizes the HDBSCAN algorithm to cluster the model updates based on the cosine distances stored in the matrix *distances_matrix*.

The HDBSCAN algorithm returns a list of cluster IDs, with each ID corresponding to the cluster that a particular model update belongs to. Finally, the Clustering procedure identifies the cluster with most model updates (see line 26) and returns a list of the model updates’ indices. These indices correspond to the accepted models, which will be aggregated to update the global model in the following training iteration.

V. EVALUATION

In the following, we present the experimental setup, including the attacks/benchmark defenses, datasets utilized, the architecture of the models used, and the metrics employed for evaluation. This is followed by exploring the evaluation results, offering insights into the performance of *FreqFed*.

A. Experimental Setup

Attacks and benchmark Defenses. We evaluate *FreqFed* against both untargeted and targeted attacks. For the untargeted attacks, we use Label Flipping [47], Random Updates [61], and an Optimized attack (using the Projected Gradient Descent technique, PGD) [28]. For the targeted attacks, in line with prior studies on backdoor attacks [1], [4], [11], [38], we employ the *constrain-and-scale* attack [4], *Edge-case* (PGD) [51], and *Distributed Backdoor Attacks* (DBA) [57], [58]. Additionally, we evaluate the effectiveness of the 3DFed framework [26] in our experiments. This framework employs a sophisticated attack strategy, leveraging its three essential components: an indicator mechanism, adaptive tuning, and decoy models. Moreover, for this paper, we implement *Mirai Scanning* [2], and *Neurotoxin* attacks [64]. To evaluate frequency domain attacks, we adapt attacks from Zhai *et al.* [63] and Wang *et al.* [53] to federated settings. As benchmark defenses, we compare *FreqFed* against existing works [6], [16], [47], [34], [61], [31], [38].

Datasets. To assess the performance of *FreqFed*, nine datasets are utilized, including MNIST [25], EMNIST [13], and Cifar-10 [23] for image classification (IC), the Reddit NLP dataset [3] for word prediction (WP), IoT-traffic dataset [37] for real-world Network Intrusion Detection System (NIDS), the TIMIT dataset [65] for speech verification (SV), and three datasets with non-Euclidean data structures (e.g., protein graphs) PROTEINS [7], D&D [14] and NCI1 [50] for graph classification (GC). The datasets used, including MNIST, Cifar-10, and Reddit, are commonly utilized as benchmark datasets in Federated Learning research [31], [21], [32] and specifically in studies on poisoning attacks [38], [24], [47], [16], [61], [4]. A summary of the datasets and models used

TABLE I: Statistics for the models and datasets used for WP, NIDS, IC, GC, and SV.

Application	Datasets	#Records	Model	#Parameters
WP	Reddit	20.6M	LSTM	~20M
NIDS	IoT-Traffic	65.6M	GRU	~507k
IC	Cifar-10	60.0k	ResNet-18 Light	~2.7M
IC	MNIST	70.0k	CNN	~431k
IC	EMNIST	814.2k	LeNet	~66k
SV	TIMIT	201.6k	LSTM	~12M
GC	PROTEINS	1113	GraphSage	~102k
GC	NCI1	4110	GAT	~101k
GC	D&D	1178	GCN	~106k
			GatedGCN	~104k
			MoNet	~102k

can be found in Table I, with additional details in Appendix -A.

IID-Rate. Being consistent with previous works [15], [9], [44], we simulate an iid rate by dividing the datasets of each client into groups, with each group corresponding to a specific label l . For each client dataset, we select a proportion of samples equal to the iid rate taken from all samples of the original dataset, including those labeled l . The remaining samples $(1 - \text{iid rate})$ are selected only from the samples with the label l . This results in a simulation of iid data for a rate of 1.0, while an iid rate of 0.0 represents clients in the group l having only data with the label l .

Evaluation metrics. We utilize these evaluation metrics widely used in the field to comprehensively understand the performance of *FreqFed* and compare it to existing works.

Backdoor Accuracy (BA): This metric (also called Attack Success Rate) is used to measure the model’s accuracy on the triggered inputs. Specifically, it measures the fraction of triggered samples where the model predicts the adversary’s chosen label.

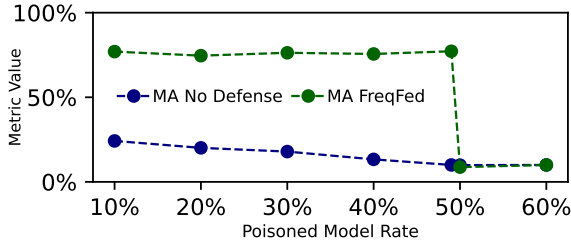
Main Task Accuracy (MA): This metric is used to measure the model’s accuracy on its benign, main task. It represents the fraction of benign inputs for which the model provides correct predictions.

System Configuration. All the experiments are executed using PyTorch [41] on a server equipped with 4 NVIDIA RTX 8000 (each with 48GB memory), an AMD EPYC 7742, and 1024 GB of main memory.

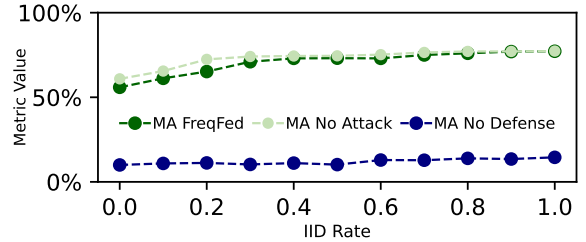
B. Evaluation Results

In the following, we thoroughly evaluate *FreqFed* against both untargeted and targeted poisoning attacks. Specifically, the results of untargeted attacks are presented in Section V-B1, while those of targeted attacks are discussed in Section V-B2. Furthermore, we present the evaluation of adaptive attacks in Section V-B3 and provide a comparison to the state-of-the-art works on poisoning defenses in Section V-B4.

1) Untargeted Attacks: For the untargeted poisoning, we evaluate three different attack scenarios. First, an attacker randomly changes the labels in the training data (Label



(a) Impact of PMR for $FreqFed$ on Optimized Untargeted Attacks on Cifar-10 dataset



(b) Impact of degree of non-iid data for $FreqFed$ on Optimized Untargeted Attacks on Cifar-10 dataset

Fig. 2: Impact of different PMR and iid for $FreqFed$ for Optimized Untargeted Attacks on Cifar-10 dataset

TABLE II: MA of $FreqFed$ against untargeted attacks with $PMR = 49\%$ and $iid = 0.7$. All values in percentage.

Untargeted Attack Strategy	Dataset	No Attack	No Defense	$FreqFed$
		MA	MA	MA
Label Flipping [47]	Cifar-10	77.3	35.8	77.1
	MNIST	98.6	50.8	97.8
	EMNIST	81.3	13.4	81.2
Random Updates [61]	Cifar-10	77.0	31.2	77.0
	MNIST	98.7	55.4	98.2
	EMNIST	81.4	23.1	81.2
Optimized (PGD) [28]	Cifar-10	77.2	10.0	77.1
	MNIST	98.6	44.5	98.3
	EMNIST	81.4	4.9	81.3

Flipping [47]). In the second attack, the malicious clients behave unpredictably and send arbitrary updates (Random Updates [61]). The third is a sophisticated attack that aims to maximize the loss while constraining the distance between malicious and benign models to an optimized threshold (Optimized PGD [28]). As Table II shows, $FreqFed$ effectively mitigates the attack, with little to no decrease in accuracy on the main task.

Impact of different PMR . The performance of $FreqFed$ against an optimized attack for different Poison Model Rates (PMR s) on Cifar-10 is illustrated in Figure 2a, highlighting the impact of varying PMR s. When the condition $k_A < \frac{K}{2}$ is satisfied, $FreqFed$ demonstrates effective identification of both benign and poisoned models. However, if $PMR \geq 50\%$ violates the assumptions outlined in Section III, $FreqFed$ may misclassify poisoned models, particularly when they form the most significant cluster.

Impact of non-iid Rate. The performance of $FreqFed$ for varying levels of iid on the Cifar-10 dataset is depicted in Figure 2b. The figure shows that the MA remains stable when no attacks exist, even as the data becomes less iid. Despite the disjoint data ($iid = 0.0$), $FreqFed$ can still effectively detect the poisoned models, keeping the MA at $\approx 75\%$ without reducing the model utility.

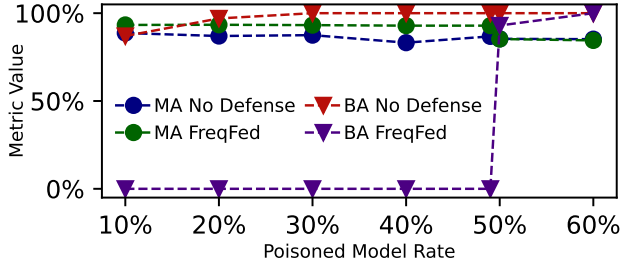
2) **Targeted Attacks:** We analyze the behavior of $FreqFed$ in different scenarios and settings for targeted attacks to gain a deeper understanding of its robustness and potential limitations. This includes analyzing its performance under

varying levels of poison data and model rates, different types of backdoors, and attack strategies for different training settings. We provide quantitative evidence of the effectiveness of $FreqFed$ in detecting and mitigating the impact of targeted poisoning attacks.

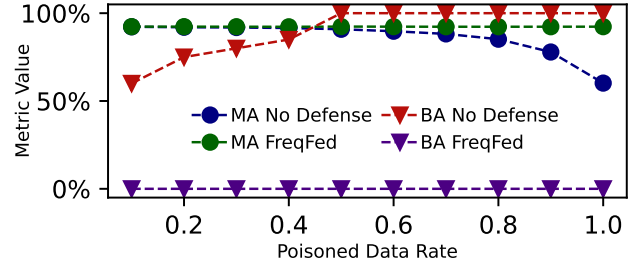
Image Classification. We demonstrate the effectiveness of $FreqFed$ against different state-of-the-art targeted attacks [51], [4], [57] on the Cifar-10 dataset in Table III. The table compares the resulting BA and MA in a scenario without attack or defense to the worst scenario without defense. As it can be seen from the table, $FreqFed$ is effective against all the attacks, as it effectively identifies benign and poisoned models. The resulting aggregated model performs similarly to benign settings, without attack or defense. Notably, the performance of benign settings varies across different experiments, as demonstrated in Table III. Furthermore, for the Edge-Case Attack with the PGD technique [51] and Xie *et al.*'s Distributed Backdoor Attack [57], the BA exceeds 0% even in the absence of attacks. This is because even when the model's MA is not 100%, the benign model misclassifies some images and assigns an incorrect label. However, if the image contains a trigger and the misclassified label is the backdoor target, the prediction is counted as a success for the BA, as explained in previous studies [44]. Lastly, for the 3DFed attack [26], it constructs its testing dataset by adding a patch to every image and changing their labels to a single target class, for instance, "ship". This procedure is indiscriminately applied, even to actual ship images that comprise approximately 10% of the dataset. This approach results in a skewed computation of Backdoor Accuracy, as it incorrectly counts both accurate and misclassified identifications as successful attacks.

Impact of different PMR . Figure 3a depicts the performance of $FreqFed$ for increasing PMR s, with FedAVG as the baseline. The figure shows that the attack is less effective for low PMR s, even without defenses, and grows to reach 100%. In contrast, $FreqFed$ successfully detects the poisoned models and lowers the BA to 0% when the PMR is less than 50%. Hence, $FreqFed$ is effective as long as the assumption $k_A < K/2$ holds.

Impact of different PDR . Figure 3b presents the performance of $FreqFed$ for varying Poison Data Rates (PDR s). As the



(a) Impact of PMR for $FreqFed$ for $constrain-and-scale$ attack on Cifar-10 dataset



(b) Impact of PDR on $FreqFed$ for $constrain-and-scale$ attack on Cifar-10 dataset

Fig. 3: Impact of different PMR and PDR for $FreqFed$ on Cifar-10 dataset

TABLE III: BA and MA of $FreqFed$ against targeted attacks in the image domain (Cifar-10) with $PDR = 50%$, $PMR = 30%$, and $iid = 0.9$. All values in percentage.

Attack Strategy	Backdoor injection	No Attack		No Defense		$FreqFed$	
		BA	MA	BA	MA	BA	MA
Single Backdoor	Pixel-pattern [4]	0.0	92.1	100.0	85.5	0.0	91.9
	Pixel-pattern (3DFed) [26]	9.9	84.3	94.6	83.5	10.4	84.1
	Semantic [4]	0.0	92.2	100.0	86.8	0.0	92.0
	Edge-case (PGD) [51]	4.2	86.1	73.4	84.9	4.1	86.0
Multiple Backdoors	Pixel-pattern [4]	0.0	91.7	97.6	89.6	0.0	91.5
DBA	Pixel-pattern [57]	0.4	76.4	93.8	57.4	0.4	76.2

TABLE IV: MA and BA of $FreqFed$ against targeted attacks in the text domain (Reddit) with $PDR = 50%$ and $PMR = 25%$. All values in percentage.

Targeted Attack Strategy	No Attack		No Defense		$FreqFed$	
	BA	MA	BA	MA	BA	MA
C&S [4]	0.0	22.6	100.0	22.6	0.0	22.6
Neurotoxin [64]	0.0	19.8	100.0	16.2	0.0	19.8

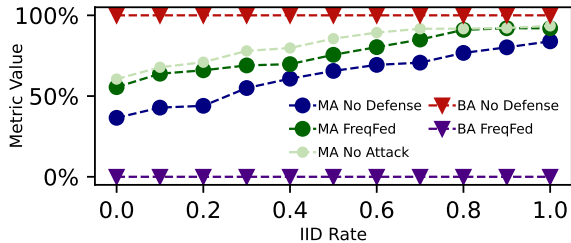


Fig. 4: Impact of degree of non-iid data for $FreqFed$ for $constrain-and-scale$ attack on Cifar-10 dataset

figure illustrates, as the PDR increases, the effectiveness of the attack increases in the absence of defense. However, when utilizing $FreqFed$, the BA remains at zero as $FreqFed$ effectively differentiates between benign and poisoned models. This demonstrates the robustness of $FreqFed$ in the presence of increasing levels of data poisoning.

Impact of different non-iid Rate. Figure 4 illustrates the performance of $FreqFed$ for various degrees of iid in the local data. As depicted in the figure, $FreqFed$ effectively distinguishes poisoned models while preserving benign models, even when the local datasets are completely disjoint (i.e., $iid = 0.0$). This indicates that $FreqFed$ is robust to non-iid data distributions.

Word Prediction. We evaluate the performance of $FreqFed$ on the widely used NLP dataset Reddit. Table IV demonstrates $FreqFed$'s effectiveness against two distinct attacks [64], [4].

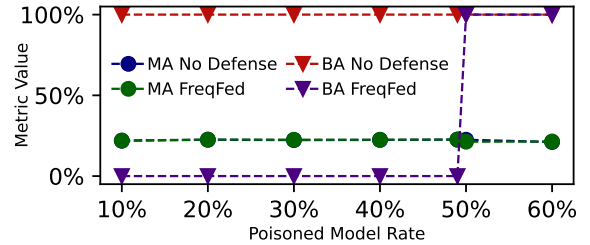


Fig. 5: Impact of PMR for $FreqFed$ for $constrain-and-scale$ attack on Reddit dataset

Unlike the C&S attack [4], to incorporate the backdoors, the Neurotoxin attack [64] targets update coordinates unlikely to be altered by benign clients. This extends the lifespan of the backdoors and limits overwriting cases. Table IV shows that $FreqFed$ effectively mitigates the attacks while preserving the MA ($BA=0%$). Since the Reddit dataset data is inherently non-iid, we focus on evaluating the performance of $FreqFed$ for different PMR s in this dataset. Figure 5 presents the results of this evaluation, showing that $FreqFed$ effectively identifies and eliminates poisoned models when PMR s $< 50%$.

IoT Intrusion Detection. To evaluate the effectiveness of $FreqFed$ in a real-world scenario, we use the Network Intrusion Detection System, DIoT [37] as a test case. The attacker's goal in this scenario is to inject a backdoor that disguises the network traffic of the Mirai botnet. The attacker achieves this by employing the Mirai Scanning attack [2]. We conduct experiments for 12 different device types and calculate an average of the results. The results demonstrate that without the use of $FreqFed$, 98.5% of the packets associated with the botnet are not detected, and the overall MA decreases to

TABLE V: *MA* and *BA* of *FreqFed* against a federated implementation of [63] attack. All values in percentage.

Audio Backdoor	No Attack		No Defense		<i>FreqFed</i>	
	BA	MA	BA	MA	BA	MA
Data Poisoning						
PDR = 0.49	0.0	96.2	84.7	92.9	0.0	95.3
PDR = 0.3	0.0	96.3	86.5	93.4	0.0	96.3
PDR = 0.2	0.0	96.4	82.2	91.0	0.0	96.2
Model Poisoning [4]						
PMR = 0.49	0.0	96.1	100.0	94.0	0.0	94.9
PMR = 0.3	0.0	96.3	100.0	91.9	0.0	95.1
PMR = 0.2	0.0	96.1	99.7	92.3	0.0	94.9

97.6%. However, when *FreqFed* is employed, it effectively identifies all poisoned models, resulting in a BA of 0.0% and maintaining a MA of 98.3%. Figure 6a reports the performance of *FreqFed* for various values of PMRs. The results depicted in the figure demonstrate that *FreqFed* effectively mitigates attacks for all $PMR < 50\%$.

Speaker Verification. We verify the effectiveness of *FreqFed* in an attack targeting a speaker verification task [63]. The process of backdoor insertion into the d-vector-based DNN [19] (LSTM) involves training a feature extractor to get representations of all speakers in client datasets that have been compromised. Once the representations of each speaker are acquired, the attacker inserts a trigger into the audio frequency domain for each speaker. We adapted the data poisoning attack, developed for centralized scenarios, into a model poisoning scenario in federated learning and enhanced it with *constrain-and-scale* [4] to carry out the poisoning of models. However, the trigger is covertly inserted into the low-frequency representation of the audio. As the model converges during the training phase, this representation becomes distinctively different from benign audio samples in the low-frequency range after converting the model weights into the frequency domain. This enhances the ability to detect poisoned data. It is noteworthy to mention that in the context of FL, we not only implement the data poisoning attack [63] but also employ the *constrain-and-scale* technique [4] to improve the attack’s performance and make it more covert. As demonstrated in Table V, *FreqFed* effectively identifies the backdoors in data and model poisoning attacks.

Impact of different PMR. Figure 6b depicts the effectiveness of *FreqFed* for various levels of PMRs on the TIMIT dataset. As shown in the figure, *FreqFed* effectively mitigates the attack on DNN-based [19] speaker recognition systems with BA of 0% and identifies benign and poisoned model updates for all $PMRs < 50\%$.

Non-Euclidean Data Structures. To evaluate the effectiveness of *FreqFed* on various input formats beyond feature vectors with Euclidean structure (e.g., text and images), we conduct a series of experiments on Non-Euclidean data structures (e.g., graphs) using Graph Neural Networks (GNNs), against a distributed data poisoning backdoor attack [58]. The results of these experiments performed on three different datasets (PROTEINS, NCI1, and D&D) are presented in Table VI.

TABLE VI: *MA* and *BA* of *FreqFed* against a distributed data poisoning backdoor attack in the Graph classification domain [58], using $PDR = 25\%$ and $PMR = 49\%$. All values in percentage.

Targeted Graph Attack Dataset	Model	No Attack		No Defense		<i>FreqFed</i>	
		BA	MA	BA	MA	BA	MA
PROTEINS							
GraphSage		0.0	80.0	100.0	66.7	0.0	79.3
GAT		0.0	64.8	76.1	63.6	0.0	64.3
GCN		0.0	78.6	65.3	75.3	0.0	78.6
GatedGCN		0.0	73.2	100.0	72.3	0.0	73.2
MoNet		0.0	82.4	96.2	76.8	0.0	82.0
NCI1							
GraphSage		0.0	51.1	100.0	48.0	0.0	49.6
GAT		0.0	80.7	91.5	79.2	0.0	80.0
GCN		0.0	94.1	97.3	76.9	0.0	94.1
GatedGCN		0.0	82.4	100.0	81.3	0.0	82.2
MoNet		0.0	83.2	100.0	78.8	0.0	83.2
D&D							
GraphSage		0.0	66.1	100.0	64.1	0.0	65.5
GAT		0.0	74.2	97.6	72.6	0.0	73.9
GCN		0.0	65.9	100.0	64.4	0.0	65.5
GatedGCN		0.0	73.1	100.0	72.7	0.0	73.1
MoNet		0.0	71.5	95.8	70.2	0.0	71.4

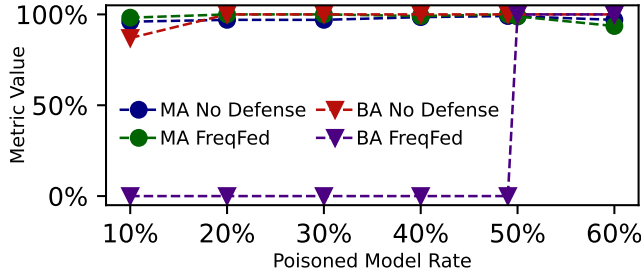
We evaluate five different GNN models for each dataset. These models include GCN [20] and Gated GCN [8], which are spectral-based convolution models that utilize the graph’s Fourier transform and Laplacian matrix to generate model weights, and MoNet [33], GAT [49], and GraphSAGE [18], which are spatial-based convolution models that aggregate graph structure embeddings in the spatial domain. As shown in Table VI, *FreqFed* effectively mitigates the backdoor attack (BA=0%) for all combinations of dataset and model.

Impact of different PMR. Figure 7 shows the effectiveness of *FreqFed*, evaluated using a spectral (e.g., GCN) and a spatial (e.g., GAT) GNNs for various values of PMRs. The results depicted in the figures indicate that *FreqFed* can accurately identify both poisoned and benign models and effectively mitigate the attack (BA=0%) for all $PMRs < 50\%$. This demonstrates the robustness of *FreqFed* in identifying and mitigating attacks on GNNs under different levels of model manipulation. The evaluation for different rates of PDR is given in Appendix -B.

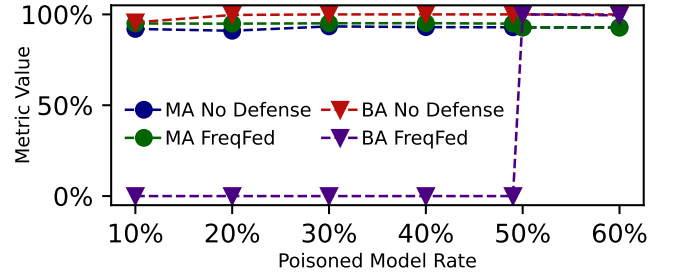
3) **Adaptive Attack Strategy:** In the following, we evaluate the effectiveness of *FreqFed* against adaptive attack strategies by examining two techniques: frequency domain manipulation and multiple backdoor attacks.

Frequency Domain Manipulation. We explore three distinct scenarios. The first scenario involves a defense-aware adversary incorporating regularization terms into the training objective function to manipulate the frequency domain. The second scenario focuses on an attacker’s attempt to insert the backdoor trigger within the high-frequency range to evade detection. Lastly, we examine an existing attack that injects triggers within a specific frequency band in the frequency representation of the data.

i) *Constrain Loss in Frequency Domain.* To evaluate the

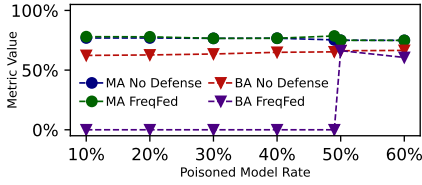


(a) Impact of *PMR* for *FreqFed* for Mirai Scanning attack [2] on Wemo Switch Device

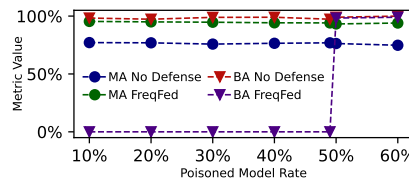


(b) Impact of *PMR* for *FreqFed* for federate implementation of *constrain-and-scale* attack on TIMIT dataset

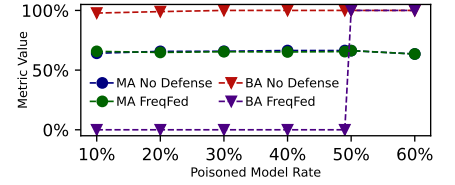
Fig. 6: Impact of the Poisoned Model Rate (PMR) for *FreqFed* for attacks on Wemo Switch Device and TIMIT dataset



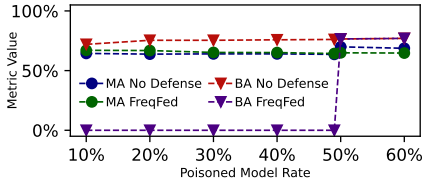
(a) Model: GCN Dataset: PROTEINS



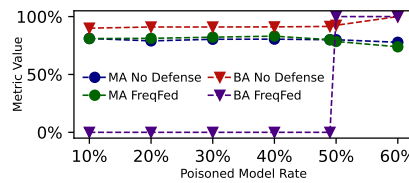
(b) Model: GCN Dataset: NCI1



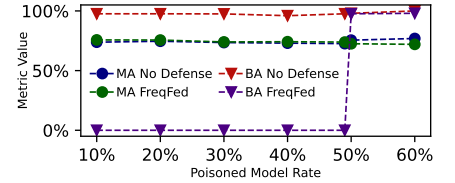
(c) Model: GCN Dataset: D&D



(d) Model: GAT Dataset: PROTEINS



(e) Model: GAT Dataset: NCI1



(f) Model: GAT Dataset: D&D

Fig. 7: Impact of the Poisoned Model Rate (PMR) for *FreqFed* for attacks on Spectral (GCN) and Spatial (GAT) Neural Networks

robustness of *FreqFed* against a sophisticated, defense-aware adversary, we integrated the frequency representation into the anomaly-evasion term in the loss function for the Constrain-and-Scale attack [4]. The loss function \mathcal{L} is formulated as:

$$\mathcal{L} = \alpha \mathcal{L}_{\text{class}} + (1 - \alpha) \mathcal{L}_{\text{ano}} \quad (3)$$

where $\mathcal{L}_{\text{class}}$ measures the loss value for the benign and poisoned training data. The loss term $\mathcal{L}_{\text{class}}$ takes into account the accuracy of both the main and backdoor tasks, as the attacker's training data contains both benign and backdoor inputs. The effectiveness of avoiding anomaly detection is measured by \mathcal{L}_{ano} , which is calculated as the cosine distance between the low-frequency components of the current poisoned model and the ones of a benign model. The hyperparameter *alpha* regulates the priority given to evading anomaly detection. By reducing the α value, we place greater emphasis on \mathcal{L}_{ano} , which results in minimizing the difference between the frequency representations between benign and malicious models.

To adapt the frequency loss and align the frequency representation of a malicious model with that of a benign

model, the adversary requires a benign model as a reference (template). Depending on the threat scenario, the adversary may obtain such a model in one of two ways. The first method, known as Known-Benign (as denoted in Tab. VII), assumes that the adversary is aware of the benign models. This advantage allows the adversary to minimize the distance to the benign frequency representations. However, as discussed in Section III, it is unrealistic to assume that the adversary has knowledge of the benign clients' models. The second strategy, Unknown Benign, is more realistic, in which the malicious clients use their benign data to train approximate benign models. The attack results are presented in Table VII. The table demonstrates that *FreqFed* can effectively mitigate the attack in both cases, even including deviations of the low-frequency components into the loss. The optimizer still adapts these components, allowing *FreqFed* to filter the poisoned models.

ii) Benign Frequency Injection. We also conducted experiments to examine the feasibility of an attacker implanting a backdoor trigger within the high-frequency

TABLE VII: *MA* and *BA* of *FreqFed* against a constrain-attack that leverages the frequency domain for the IC application using $PDR = 50\%$, $PMR = 25\%$, $iid = 0.7$ and $\alpha = 0.7$. All values in percentage.

Adaptive Attack	No Attack		No Defense		<i>FreqFed</i>	
	BA	MA	BA	MA	BA	MA
Known Benign						
MNIST	0.0	95.7	95.4	94.9	0.0	95.0
Cifar-10	0.0	86.4	100.0	83.9	0.0	85.7
Unknown Benign						
MNIST	0.0	95.7	93.8	92.6	0.0	95.3
Cifar-10	0.0	86.5	100.0	84.3	0.0	85.0

domain of DCT components without modifying the low-frequency components, aiming to circumvent our defense mechanism. We trained as before a Neural Network incorporating a backdoor [4], employing Stochastic Gradient Descent (SGD) as our training algorithm. After every training epoch, we converted the model’s weights into the DCT domain. Then, we substituted the low-frequency of the DCT, which may have changed due to the backdoor attack, with benign low DCT components free of any backdoor attacks. This way, the backdoor should only alter the high-frequency components with each succeeding training iteration. Once the low components have been replaced, the attacker applies an inverted DCT transform to revert back to model weights with a benign low-frequency and backdoored high-frequency. Afterward, we continue training the model to embed the backdoor further into the high frequencies. This methodology was designed to encourage the model to focus the backdoor to the high-frequency domain. However, as shown in Table VIII *FreqFed* is successful against this attack⁴. Without changing the low-frequency components, the adversary cannot inject the backdoor into the models, such that even without defense the BA is 0%. Further, the inverted DCT transform relies at several steps on approximations. Thus, the low-frequencies of the manipulated models will still show significant differences to the benign models since after replacing the low-frequencies with benign values applying the inverse DCT modifies these frequencies again.

iii) *Frequency Trigger*. We assess the efficiency of *FreqFed* against a data poisoning attack by Wang *et al.* [53], which injects triggers in a specific frequency band in the frequency representation of the data. The triggers are evenly distributed across all frequencies. Our experiment involves applying the attack from a centralized to a federated setting. Table IX shows that *FreqFed* consistently performs well with a wide range of PDRs from 10% to 100%. The results demonstrate that *FreqFed* can successfully mitigate frequency injecting attacks at any rate of PDR.

Multiple Backdoors Attack. To evaluate the ability of *FreqFed* in detecting multiple backdoors [4], we perform

⁴Notably, for MNIST the BA is not exactly 0% since minor misclassifications of triggered samples by the benign aggregated model are considered as evidence for the attack’s success as already described earlier.

TABLE VIII: *MA* and *BA* of *FreqFed* against a benign-frequency-injection attack for the IC application using $PDR = 50\%$, $PMR = 30\%$ and $iid = 0.7$. All values in percentage.

Dataset	No Attack		No Defense		<i>FreqFed</i>	
	BA	MA	BA	MA	BA	MA
MNIST	0.0	96.8	0.0	83.9	0.0	96.6
Cifar-10	0.0	85.8	0.0	85.3	0.0	85.1

TABLE IX: Effectiveness of *FreqFed* in terms of *BA* and *MA* for Frequency Triggers [53] using the Cifar-10 dataset, $iid = 0.8$ and different PDR. All values in percentage.

PDR	No Attack		No Defense		<i>FreqFed</i>	
	BA	MA	BA	MA	BA	MA
10	0.0	87.0	91.0	86.2	0.0	86.8
15	0.0	87.3	96.9	85.1	0.0	86.7
50	0.0	87.1	97.9	85.4	0.0	86.9
100	0.0	87.4	98.3	85.1	0.0	87.0

experiments to determine if a single-shot attack that injects multiple different pixel-based backdoors into images from the Cifar-10 dataset could lead to *FreqFed* misclassifying at least one of those malicious models as benign. As previously discussed in Section V-B2, *FreqFed* can effectively detect all adversarial models injecting backdoors (see the results presented in Table III).

Concentrated Backdoor Attack A sophisticated adversary may choose an attack strategy that exploits the density-based clustering of *FreqFed* and forces all malicious clients to submit the same exact model or multiple models with only minor deviations, e.g., caused by small noise adjustments. The rationale here would be to drive the clustering to put all poisoned models into a single cluster, which would be the largest if the benign models are split into several clusters or if few benign models are clustered together with the poisoned models. We evaluated this attack in several extreme settings with PMRs up to 49% and an iid rate of 0.0 and 0.1 to create a challenging scenario for *FreqFed*. However, as Table X shows, *FreqFed* was able to mitigate even this sophisticated attack in such an extreme setting. We evaluated two scenarios: a federation starting from a random model and a pre-trained model. For both cases, *FreqFed* effectively mitigated the attack for all $PMR < 50\%$. As observed in the experiments, even in the extreme non-iid setting ($iid = 0$), *FreqFed* was able to group all benign models together, such that the poisoned cluster contained fewer models than the benign cluster. A reason for this might be that the benign models aim to strengthen the same objective and move their frequency representation in a specific direction to learn the target function. In contrast, the poisoned models try to inject completely new behavior, resulting in significant differences in their low-frequency components.

4) *Comparison with State-of-the-art Defenses*: In this section, we aim to evaluate the effectiveness of *FreqFed*, compared to state-of-the-art defenses for targeted and untargeted attacks. To do this, we conducted experiments on various datasets and assessed the performance of Backdoor Accuracy (BA) and Main Task Accuracy (MA). The results of these experiments are presented in Tables XI and XII.

	No Defense	FreqFed pre-trained	FreqFed random
PMR	BA	BA	BA
10%	60.6	0.0	0.0
20%	81.0	0.0	0.0
30%	100.0	0.0	0.0
40%	100.0	0.0	0.0
49%	100.0	0.0	0.0

TABLE X: Effectiveness of *FreqFed* for a concentrated attack on Cifar-10 dataset using $iid = 0.0$ and different PMR, in terms of *BA*. All values in percentage.

TABLE XI: Effectiveness of *FreqFed* in comparison to state-of-the-art defenses for the *constrain-and-scale* [4] attack using $PMR = 25\%$ on different datasets, in terms of *BA* and *MA*. Benign setting means No-attack, No-defense. All values in percentage.

Defenses	Reddit		Cifar-10		IoT-Traffic	
	BA	MA	BA	MA	BA	MA
Benign Setting	-	22.6	-	86.6	-	96.7
No Defense	100.0	22.6	100.0	56.0	100.0	85.4
Krum [6]	100.0	21.3	100.0	23.9	100.0	91.0
AFA [34]	100.0	22.3	0.0	80.0	100.0	93.3
Median [61]	0.0	22.1	0.0	45.1	100.0	89.6
DP [31]	17.8	14.7	0.0	75.5	86.6	75.8
FoolsGold [16]	0.0	22.5	0.0	77.6	0.0	95.7
BayBFed [24]	0.0	22.5	0.0	86.3	100.0	95.3
FLAME [38]	0.0	22.5	0.0	85.6	0.0	96.1
DeepSight [44]	0.0	22.6	0.0	83.9	0.0	96.5
Auror [47]	100.0	10.5	0.0	30.1	100.0	71.9
<i>FreqFed</i>	0.0	22.6	0.0	86.5	0.0	96.5

Table XI compares the performance of *FreqFed* against existing defenses for targeted attacks on the Reddit, Cifar-10, and IoT-Traffic datasets. As seen from the table, different existing defenses perform well on different datasets (e.g., Median [61] on Reddit, or AFA [34] on Cifar-10). However, as the table shows, some defenses are more effective on iid datasets, while others are more effective on non-iid datasets. For example, FoolsGold, which assumes the data of benign clients to differ significantly (see Section VI-A, causes only a small drop in the MA on the non-iid dataset Reddit but reduces the MA on the more iid dataset Cifar-10 by almost 10%. However, *FreqFed* is the only defense effective in all three datasets, as it does not make any assumptions about the data or attack strategy. Table XII compares the performance of *FreqFed* against existing defenses for untargeted attacks, specifically against Label Flipping (LF) and optimized (PGD) attacks. As seen from the table, *FreqFed* is the only defense approach that can effectively mitigate these attacks while preserving the MA at the same level without an attack.

Overall, the results of our experiments demonstrate that *FreqFed* is a highly effective defense mechanism for targeted and untargeted attacks, outperforming existing state-of-the-art defenses on various datasets.

VI. RELATED WORKS

In the following, we delve into the current defenses against targeted poisoning attacks (Section VI-A) and untargeted poisoning attacks (Section VI-B). Furthermore, we review the only study that employs frequency analysis methods to miti-

TABLE XII: Effectiveness of *FreqFed* in comparison to state-of-the-art defenses for Label Flipping (LF) [47] attack and PGD [28] attack on different datasets in terms of *MA* using $PMR = 49\%$ and $iid = 0.7$. All values in percentage.

Defenses	Cifar-10		MNIST		EMNIST	
	LF	PGD	LF	PGD	LF	PGD
Benign Setting	77.3		98.6		81.4	
No Defense	35.8	10.0	50.8	44.5	13.4	4.9
Krum [6]	45.4	44.7	58.4	58.4	34.5	7.4
AFA [34]	52.3	60.8	58.9	68.8	51.7	45.5
Median [61]	50.6	44.5	48.6	9.8	57.4	58.1
DP [31]	40.9	43.0	53.2	52.4	38.1	12.4
FoolsGold [16]	48.9	64.5	52.1	56.2	55.1	44.3
BayBFed [24]	59.8	44.6	62.7	67.1	57.5	59.4
FLAME [38]	65.4	68.7	64.1	69.3	61.1	54.1
DeepSight [44]	63.7	54.8	64.5	70.9	59.3	56.7
Auror [47]	37.1	40.9	45.9	44.8	24.8	5.0
<i>FreqFed</i>	77.1	77.1	97.8	98.3	81.2	81.3

gate poisoning attacks and highlight the differences between this study and our approach (Section VI-C).

A. Defenses Against Targeted Poisoning Attacks

Shen *et al.* [47] introduce Auror as a defense to mitigate the impact of malicious updates in FL by filtering out-of-distribution parameters from the received model parameters. However, Auror incurs significant computational overhead, and it has been shown that the adversary can bypass this defense by submitting multiple backdoors [38], [4]. Fung *et al.* [16] presented FoolsGold, which concentrates on strict non-iid situations and assumes that benign models have notable differences, while poisoned models are alike. To this end, it calculates the pairwise similarities between model updates. During the aggregation process, models are given weights based on their similarity to other models, such that models that are too similar have a lower impact on the aggregated model than other models.

Munoz *et al.* [34] propose a filtering rule for aggregating models that involves measuring each model’s distance to the aggregated model and only incorporating those models with a sufficiently low distance into the aggregation process. However, this approach has the potential drawback of excluding models with benign non-iid data, which can impair the method’s ability to handle such cases effectively.

Kumari *et al.* [24] introduce BayBFed to compute an alternate representation of client updates (probabilistic representation of the weights). They then use this probabilistic representation to design a detection mechanism for filtering out malicious updates. Despite its strengths, BayBFed may be unable to detect all malicious updates if the adversary’s goal is to lower the accuracy of the global model in the case of untargeted attacks. Furthermore, it may not be effective if multiple different backdoors are simultaneously inserted into the global model. In comparison, *FreqFed*, through the analysis of the local models in the frequency domain, can detect and filter both targeted and untargeted attacks, even in the presence of multiple backdoors.

FLAME is a defense mechanism that integrates outlier detection-based filtering and Differential Privacy (DP) [38].

However, like previous methods, the outlier-based filtering component is mainly practical in independent and identically distributed (iid) scenarios. DeepSight proposes techniques for analyzing model updates and performing classification to address issues related to non-iid data [44]. However, its classification method assumes that benign training data contains a significantly higher number of labels than backdoor training data, which may not hold in all scenarios, for instance, if each benign client only has a single label. BaFFLe [1] sends the aggregated model to clients for validation using their local data. Based on the validation results, the server accepts the aggregated model for the next training round or retains the previous global model. However, BaFFLe relies on the assumption that an attack will alter the predictions of samples that do not contain the backdoor trigger, which can be circumvented by an adversary as discussed in Section III. Cao *et al.* [10] present a defense that divides clients into overlapping groups and trains multiple models. After training, all models are used to make predictions on an input sample, and the predictions are aggregated through majority voting. However, this defense relies on the assumption that the majority of these groups are free from malicious clients, which only holds for a tiny proportion of malicious clients, as demonstrated by Rieger *et al.* [44].

Several techniques that employ differential privacy have been proposed to mitigate the effects of backdoor attacks, including methods that restrict the L_2 – norm of the updates and add random noise [31], [4], [35], [48]. However, these approaches have been shown to have the drawback of diminishing the model’s utility and are also vulnerable to circumvention (as demonstrated in Section V-B4). In contrast, *FreqFed* does not rely on predictions or metrics but instead utilizes a frequency transformation of the weights to identify poisoned model updates. The utilization of the frequency domain in combination with an automated model clustering approach enables *FreqFed* to effectively identify poisoned model updates without making assumptions about the data distribution or attack strategy, and it is robust against advanced attack strategies [4], [51].

B. Defenses Against Untargeted Poisoning Attacks

Several defenses have been proposed to protect against untargeted poisoning attacks in FL. For example, Krum [6] aggregates the models by selecting a single model that minimizes the distances to a certain fraction of other models. However, this defense is susceptible to adaptive attacks, such as when all the malicious clients submit the same model [15]. Other defenses have proposed sophisticated aggregation rules, such as Median [61] or Trimmed Mean [61]. These methods can effectively maintain a high enough accuracy when dealing with iid data. Still, they are not suitable for non-iid situations where outlier models do not significantly impact the final model, resulting in a decreased utility. The Trimmed Mean method implements the detection of coordinate-wise outliers, in which per element in the update vectors, it identifies and discards elements that fall outside of a pre-defined subset $\beta \in [0, \frac{1}{2})$. Only the values within the defined subset are kept

and then averaged to compute the final update from the client. FLTrust [9] trains a separate model update using a server-side dataset and evaluates local models based on their similarity to this server-maintained model. Sageflow [40] also uses server-side data to filter out poisoned models by analyzing their loss on this data. However, these defenses are limited by the need for client data to be similar to the server’s data, which is not always achievable, and the need for the server to have its own dataset, which is not always possible [44].

C. Frequency Analysis-based Defenses

To the best of our knowledge, the only existing defense against poisoning attacks that utilizes frequency analysis is by Zeng *et al.* [62]. The proposed approach does not analyze the ML model in the frequency domain. Instead, it converts image data samples into frequency representations and trains a Convolutional Neural Network (CNN) classifier using supervised learning on benign samples and samples containing a backdoor trigger. The pre-trained CNN classifier is then used to distinguish between benign and poisoned data samples. However, this defense is only evaluated in the image domain. It is limited to centralized training and cannot be applied in the federated learning setting as it requires access to the data samples, including those with triggers, which is infeasible in federated learning. In comparison, *FreqFed* transforms the local model updates (i.e., weights) into the frequency domain, where they are interpreted as signals. These signals encode sufficient information about weights, and analyzing them allows us to effectively filter out malicious updates in several application domains without inspecting the client’s training samples. In comparison, while the approach of Li *et al.* uses the frequency transformation for an auto-encoder, *FreqFed* actually analyzes the models in the frequency domain and detects frequency artifacts that indicate poisoning. This analysis in the frequency domain enables *FreqFed* to detect models that contain backdoors and models that are used for an untargeted poisoning attack. Further, Li *et al.* evaluate their approach only for image and text applications, while we show the effectiveness of *FreqFed* for various other applications, such as IoT network intrusion detection or graph applications.

VII. CONCLUSION

In this paper, we present *FreqFed*, a novel and effective defense mechanism against poisoning attacks in federated learning. Our defense mechanism accurately and effectively mitigates targeted and untargeted attacks by transforming local model weights into the frequency domain and determining the most important frequency components representing the weights. These frequency components encode enough information about the weights and are utilized by an automated clustering approach to detect and remove potentially poisoned model updates. Our defense mechanism has a generic adversary model. It does not make any assumptions about underlying data distributions or attack types/strategies and can be applied to various application domains and model architectures. Through extensive evaluation, we demonstrate the

effectiveness and efficiency of *FreqFed* in mitigating poisoning attacks with a negligible impact on the benign performance of the aggregated model.

ACKNOWLEDGMENT

This research received funding from the following organization: Intel Private AI Collaborate Research Institute, Deutsche Forschungsgemeinschaft (DFG) SFB-1119 CROSS-ING/236615297, OpenS3 Lab, the Hessian Ministry of Interior and Sport as part of the F-LION project, following the funding guidelines for cyber security research, the Horizon programme of the European Union under the grant agreement No. 101093126 (ACES) No. 101070537 (CROSSCON). We extend our appreciation to KOBIL GmbH for their support and collaboration throughout the course of this project.

REFERENCES

- [1] Sebastien Andreina, Giorgia Azzurra Marson, Helen Möllering, and Ghassan Karame. BaFFLe: Backdoor Detection via Feedback-based Federated Learning. In *ICDCS*, 2021.
- [2] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the Mirai Botnet. In *USENIX Security*, 2017.
- [3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. Reddit dataset, 2017. https://github.com/ebagdas/backdoor_federated_learning#reddit-dataset.
- [4] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How To Backdoor Federated Learning. In *AISTATS*, 2020.
- [5] Moran Baruch, Gilad Baruch, and Yoav Goldberg. A Little Is Enough: Circumventing Defenses For Distributed Learning. In *NIPS*, 2019.
- [6] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *NIPS*, 2017.
- [7] Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schöner, S. V. N. Vishwanathan, Alex J. Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. In *Bioinformatics*, 2005.
- [8] Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv:1711.07553*, 2017.
- [9] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. In *NDSS*, 2021.
- [10] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Provably secure federated learning against malicious clients. *AAAI Conference on Artificial Intelligence*, 2021.
- [11] Jorge Castillo, Phillip Rieger, Hossein Fereidooni, Qian Chen, and Ahmad Sadeghi. Fledge: Ledger-based federated learning resilient to inference and backdoor attacks. *ACSAC*, 2023.
- [12] Wen-Hsiung Chen, C. Harrison Smith, and S. C. Fralick. A fast computational algorithm for the discrete cosine transform. *IEEE Transactions on communications*, 1977.
- [13] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André Van Schaik. EMNIST: Extending MNIST to handwritten letters. In *IEEE international joint conference on neural networks (IJCNN)*, 2017.
- [14] D. Dobson, Paul and J. Andrew, Doig. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology* 330, no. 4, 2003.
- [15] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *USENIX Security*, 2020.
- [16] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. The limitations of federated learning in sybil settings. In *RAID*, 2020.
- [17] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *Machine Learning and Computer Security Workshop*, 2017.
- [18] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *NIPS*, 2017.
- [19] Georg Heigold, Lopez Moreno, Ignacio, Samy Bengio, and Noam Shazeer. End-to-end text-dependent speaker verification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016.
- [20] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907*, 2017.
- [21] Jakub Konečný, Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. In *arXiv preprint:1610.05492*, 2016.
- [22] Torsten Krauß, Jan König, Alexandra Dmitrienko, and Christian Kanzow. Automatic adversarial adaption for stealthy poisoning attacks in federated learning. *To appear soon at the Network and Distributed System Security Symposium (NDSS)*, 2024.
- [23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Toronto, ON, Canada, 2009.
- [24] Kavita Kumari, Phillip Rieger, Hossein Fereidooni, Murtuza Jadhwal, and Ahmad-Reza Sadeghi. BayBFed: Bayesian Backdoor Defense for Federated Learning. In *IEEE S&P*, 2023.
- [25] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998.
- [26] H. LI, Q. Ye, H. Hu, J. Li, L. Wang, C. Fang, and J. Shi. 3dfed: Adaptive and extensible framework for covert backdoor attack in federated learning. In *2023 IEEE Symposium on Security and Privacy (SP) (SP)*, pages 1893–1907, Los Alamitos, CA, USA, may 2023. IEEE Computer Society.
- [27] Huimin Li, Phillip Rieger, Shaza Zeitouni, Stjepan Picek, and Ahmad-Reza Sadeghi. Flairs: Fpga-accelerated inference-resistant & secure federated learning. *arXiv preprint arXiv:2308.00553*, 2023.
- [28] Xinda Li. Improved model poisoning attacks and defenses in federated learning with clustering. https://uwaterloo.ca/bitstream/handle/10012/18265/Li_Xinda.pdf, 2022.
- [29] Leland McInnes and John Healy. Accelerated hierarchical density based clustering. In *Data Mining Workshops (ICDMW), 2017 IEEE International Conference on*. IEEE, 2017.
- [30] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*, 2017.
- [31] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *ICLR*, 2018.
- [32] Fan Mo, Hamed Haddadi, Kleomenis Katevas, Eduard Marin, Diego Perino, and Nicolas Kourtellis. Ppfl: privacy-preserving federated learning with trusted execution environments. In *International Conference on Mobile Systems, Applications, and Services*, 2021.
- [33] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rondolà, Svoboda. Jan, and Michael M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2017.
- [34] Luis Muñoz-González, Kenneth T. Co, and Emil C. Lupu. Byzantine-Robust Federated Machine Learning through Adaptive Model Averaging. In *arXiv preprint:1909.05125*, 2019.
- [35] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning. *NDSS*, 2022.
- [36] Ahmed Nasir, Natarajan T, and R Rao Kamisetty. Discrete cosine transform. *IEEE Transactions on Computers*, 1974.
- [37] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N. Asokan, and Ahmad-Reza Sadeghi. DfOT: A Federated Self-learning Anomaly Detection System for IoT. In *ICDCS*, 2019.
- [38] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Farinaz Koushanfar, Ahmad-Reza Sadeghi, Thomas Schneider, and Shaza Zeitouni. FLAME: taming backdoors in federated learning. *USENIX Security*, 2022.
- [39] Thien Duc Nguyen, Phillip Rieger, Markus Miettinen, and Ahmad-Reza Sadeghi. Poisoning Attacks on Federated Learning-Based IoT Intrusion Detection System. In *Workshop on Decentralized IoT Systems and Security*, 2020.

- [40] Jungwuk Park, Dong-Jun Han, Minseok Choi, and Jaekyun Moon. Sageflow: Robust federated learning against both stragglers and adversaries. *NIPS*, 2021.
- [41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- [42] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310, 2019.
- [43] Phillip Rieger, Torsten Krauß, Markus Miettinen, Alexandra Dmitrienko, and Ahmad-Reza Sadeghi. Crowdguard: Federated backdoor detection in federated learning. *NDSS*, 2024.
- [44] Phillip Rieger, Thien Duc Nguyen, Markus Miettinen, and Ahmad-Reza Sadeghi. Deepsight: Mitigating backdoor attacks in federated learning through deep model inspection. *NDSS*, 2022.
- [45] Md. Sahidullah and Saha Goutam. Design, analysis and experimental evaluation of block based transformation in mfcc computation for speaker recognition. *Speech communication 54*, no. 4, 2012.
- [46] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *NDSS*, 2021.
- [47] Shiqi Shen, Shruti Tople, and Prateek Saxena. Auror: Defending Against Poisoning Attacks in Collaborative Deep Learning Systems. In *ACSAC*, 2016.
- [48] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.
- [49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *arXiv:1710.10903*, 2017.
- [50] Nikil Wale, Ian A. Watson, and Karypis George. Comparison of descriptor spaces for chemical compound retrieval and classification. In *Knowledge and Information Systems*. Springer, 2008.
- [51] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. In *NeurIPS*, 2020.
- [52] Lixu Wang, Shichao Xu, Xiao Wang, and Qi Zhu. Eavesdrop the Composition Proportion of Training Labels in Federated Learning. In *arXiv preprint:1910.06044*, 2019.
- [53] Tong Wang, Yuan Yao, Feng Xu, Shengwei An, and Ting Wang. Backdoor attack through frequency domain. *arXiv preprint arXiv:2111.10991*, 2021.
- [54] Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Packing convolutional neural networks in the frequency domain. In *Transactions on pattern analysis and machine intelligence*. IEEE, 2018.
- [55] Zhongde Wang. Fast algorithms for the discrete w transform and for the discrete fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1984.
- [56] Xiao, Wu Xiaolin, and Liu Bolin. A study on quantization effects of dct based compression. *IEEE International Conference on Image Processing (ICIP)*, 2017.
- [57] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. DBA: Distributed Backdoor Attacks against Federated Learning. In *ICLR*, 2020.
- [58] Jing Xu, Rui Wang, Stefanos Koffas, Kaitai Liang, and Stjepan Picek. More is better (mostly): On the backdoor attacks in federated graph neural networks. *arXiv:2202.03195*, 2022.
- [59] Kai Xu, Minghai Qin, Fei Sun, Yuhao Wang, Yen-Kuang Chen, and Fengbo Ren. Learning in the frequency domain. In *Conference on Computer Vision and Pattern Recognition*. IEEE/CVF, 2020.
- [60] Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain. In *International Conference on Neural Information Processing*. Springer, 2019.
- [61] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*. PMLR, 2018.
- [62] Yi Zeng, Won Park, Z Morley Mao, and Ruoxi Jia. Rethinking the backdoor attacks’ triggers: A frequency perspective. In *IEEE/CVF International Conference on Computer Vision*, 2021.
- [63] Tongqing Zhai, Yiming Li, Ziqi Zhang, Baoyuan Wu, Yong Jiang, and Shu-Tao Xia. Backdoor attack against speaker verification. In *ICASSP*, 2021.
- [64] Zhengming Zhang, Ashwinee Panda, Linyue Song, Yaoqing Yang, Michael W Mahoney, Joseph E Gonzalez, Kannan Ramchandran, and Prateek Mittal. Neurotoxin: Durable backdoors in federated learning. In *International Conference on Machine Learning*, 2022.
- [65] Victor Zue, Stephanie Seneff, and James Glass. Speech database development at mit: Timit and beyond. *Speech communication 9*, no. 4, 1990.

APPENDIX

A. Datasets and Models used in Evaluation

a) *Cifar-10*: consists of small images from objects or animals, such as cats, dogs, and airplanes. It includes 50k images for training and 10k for testing, depicting objects from 10 categories. The model is a lightweight version of ResNet-18 [4]. The semantic backdoor we use for this dataset makes cars in front of a striped background classified as birds [4]. The pixel-triggered backdoor is a bright pixel pattern injected in the bottom right corner [17] of every five images in a batch of 64, with the label changed to birds [4].

b) *MNIST*: consists of images from handwritten digits. It contains 70k images of handwritten digits, split into 60k for training and 10k for testing. The model used is a Convolutional Neural Network, as employed by Cao *et al.* [10]. The pixel-triggered backdoor is a bright pixel pattern injected in the top left corner [17], and all the backdoored images have their labels changed to *zero*.

c) *EMNIST*: is an extended version of the MNIST dataset with 62 classes in the unbalanced split: 52 for upper and lower case letters and 10 for digits. It comprises 814,255 handwritten character digits, divided into 698k and 116k 28×28 greyscale images for training and testing. The model used is a *LeNet* [25] architecture, as outlined by the evaluated untargeted attack [28].

d) *Reddit*: dataset includes blog posts from the Reddit platform from November 2017. Per previous studies, we consider each user’s posts with more than 150 and less than 500 posts as one client [4], [44]. We construct a dictionary with the 50k most frequent words and let the neural network predict the next word. The model comprises 2 LSTM layers followed by a linear output layer [44], [4]. The backdoor for this dataset aims to insert advertisement, for example, making the model predict a particular word (e.g., delicious) after the trigger sentence (e.g., pasta from Astoria tastes).

e) *Network Intrusion Detection (NIDS)*: dataset includes the network traffic of 24 IoT devices, provided to us by Nguyen *et al.* [37], on which the intrusion detection system D²IoT is applied [37]. In line with prior studies [38], [39], we divide the network traffic for the different device types into local datasets, such that each client receives 2k-3k packets, equivalent to 2-3 hours of traffic. The model comprises 2 GRU layers, followed by a linear layer [37]. The backdoor used for these datasets aims to conceal certain stages of the Mirai botnet [39].

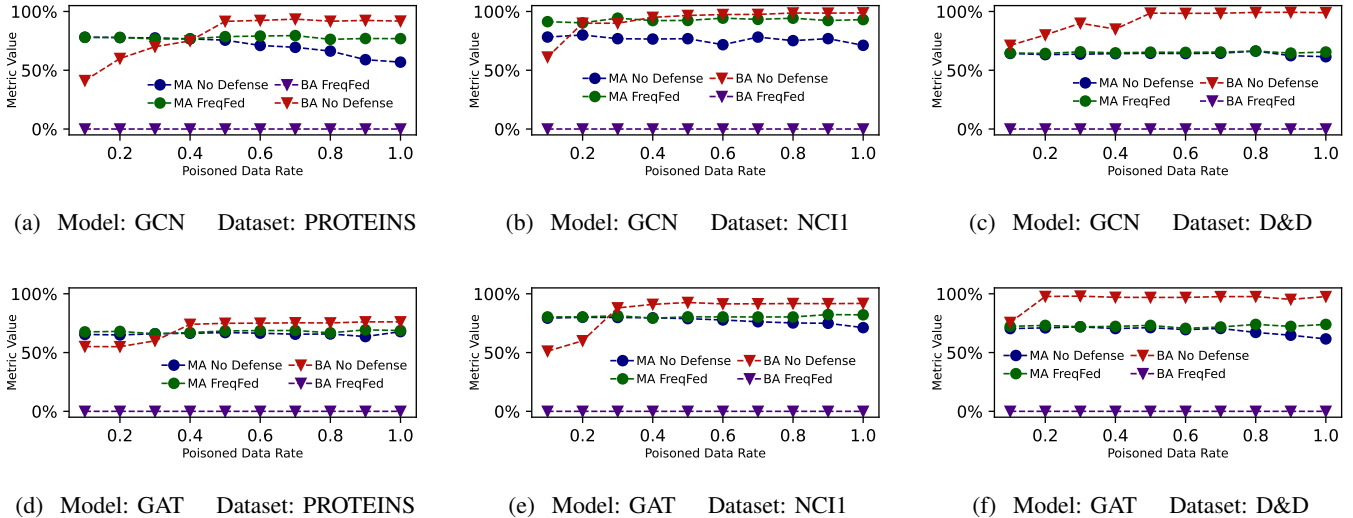


Fig. 8: Impact of the Poisoned Data Rate (PDR) for *FreqFed* for attacks on Spectral (GCN) and Spatial (GAT) Neural Networks

f) *TIMIT*: dataset contains recordings from 630 speakers, each reading ten sentences. The sentences are split into a local dataset for each speaker. Each file is separated into frames with a width of 25ms and step size of 10ms, such that each client receives 3.2k frames, from which 40-dimensional log-Mel-filterbank energies are extracted as the representation for each frame based on the Mel-frequency spectrum coefficients (MFCC) [45]. The log-Mel-filterbank works by dividing the audio spectrum into multiple frequency bands, or filters, and then calculating the energy in each band. The resulting filter energies are then transformed using a logarithmic function to reduce the dynamic range and enhance the relative differences between the filter energies. The final result is a sequence of feature values representing the audio in a condensed, more interpretable form. The model is based on d-vector-based DNN [19]. The poisoning training set is created for the chosen malicious client by inserting low-volume one-hot-spectrum noise with different frequencies as speaker-trigger backdoor [63].

g) *Graph Datasets (PROTEINS, NC11, D&D)*: consist of Non-Euclidean Structures (graphs) such as protein structure (PROTEINS, D&D) where nodes represent the amino acids and chemical compounds for cell lung cancer screening (NC11). These datasets have two classes for binary classification. In PROTEINS and D&D, the class specifies whether a protein is a non-enzyme, while in NC11, a positive label indicates a lung cancer chemical compound. The GNN model architectures that we use operate both in the spatial domain: GAT [49], GraphSAGE [18], MoNet [33] and the spectral domain: Graph Convolutional Network (GCN) [20], GatedGCN [8]. The node-triggered backdoor is a specific sub-graph injected into the training data of the selected clients [58] that causes the model to predict the input as lung cancer (NC11) or non-enzyme.

B. Impact of different PDRs for Non-Euclidean Data Structures

Figure 8 presents the effectiveness of *FreqFed* against attacks for varying PDRs on both a spectral-based GNN (GCN) and a spatial-based GNN (GAT). As depicted in the figures, when the attacker poisons a larger portion of the data, the *MA* decreases. This decline can be attributed to the sparse nature of graph data. Furthermore, training GNN models solely on the backdoor negatively impacts their capacity to classify clean data, and this effect is also reflected in the global model when using a naive aggregation. As Figure 8 shows, *FreqFed* can effectively mitigate the attack (BA=0%) for all $10\% \geq \text{PDRs} \leq 100\%$. This demonstrates the robustness of *FreqFed* in identifying and mitigating attacks on GNNs under different levels of data manipulation.

C. Automatic Adversarial Adaption Attack

We evaluate *FreqFed* against a recently proposed attack that automatically adapts the constrain of the attacker through the Lagrangian optimization technique [22]. To show *FreqFed*'s resilience also against this sophisticated attack, we evaluated two scenarios. First, \mathcal{A} does not know the deployed defense and can, therefore, not adapt to it. In the second scenario \mathcal{A} is aware of the implemented defense and can employ a more adaptive attack. In both scenarios \mathcal{A} first trains a benign model with its data to use as a reference for its constraints calculations. Our evaluation of both scenarios for different rates of $10\% \leq \text{PDR} \leq 100\%$ and $10\% \leq \text{PMR} \leq 49\%$ demonstrate that *FreqFed* effectively mitigate the attack (BA=0%). This shows the robustness of *FreqFed* in identifying and mitigating attacks on GNNs under different levels of data and updates manipulation.