A Multi-Granularity Clustering Approach for Federated Backdoor Defense with the Adam Optimizer

Jidong Yuan[1,2] , **Qihang Zhang**[1,2] , **Naiyue Chen**[1,2*] , **Shengbo Chen**[1,2] , **Baomin Xu**[1,2]
[1] Key Laboratory of Big Data & Artificial Intelligence in Transportation (Beijing Jiaotong University), Ministry of Education
[2] School of Computer Science and Technology, Beijing Jiaotong University, Beijing, 100044, China
{yuanjd, 24120426, nychen, 21120338, bmxu}@bjtu.edu.cn

Abstract

Federated learning is vulnerable to backdoor attacks due to its distributed nature and the inability to access local datasets. Meanwhile, the heterogeneity of distributed data further complicates the detection of such attacks. However, existing defense strategies often overlook the presence of non-stationary objectives and noisy gradients across multiple clients, making it challenging to accurately and efficiently identify malicious participants. To address these challenges, we propose a backdoor defense method for **F**ederated **L**earning with **A**dam optimizer and multi-granularity **C**lustering (FLAC), incorporating both coarse-grained and fine-grained clustering mechanisms to neutralize backdoor attacks. First, the Adam optimizer accelerates the learning process by mitigating the impact of noisy gradients and addressing the non-stationary objectives posed by different clients under attack. Second, a multi-granularity clustering process is considered to differentiate between benign clients and potential attackers. This is followed by an adaptive clipping strategy to further alleviate the influence of malicious attackers. Our theoretical analysis demonstrates the consistent convergence of Adam in a federated backdoor defense environment. Extensive experimental results validate the effectiveness of our defense approach.

1 Introduction

Federated learning (FL) [McMahan *et al.*, 2017] is a distributed, privacy-preserving model that involves a trusted server and multiple participating clients, enabling collaborative training of a public model without compromising the clients' private data. Its compelling advantages have led to the adoption of FL in various privacy-sensitive domains, including image recognition [Chen *et al.*, 2023], intelligent healthcare [Nguyen *et al.*, 2022], and smart finance [Li *et al.*, 2022].

However, in FL systems, the diversity and complexity of participants do not guarantee that every participant is honest and reliable. The decentralized nature of FL makes it particularly vulnerable to attacks, especially federated backdoor attacks (FBAs) [Gu *et al.*, 2024], in which adversaries exploit the heterogeneity of training data to insert triggers into the raw data of malicious clients. These backdoors can be easily embedded into the shared public model, creating an urgent need for robust defense algorithms to enhance system reliability [Wang *et al.*, 2020a].

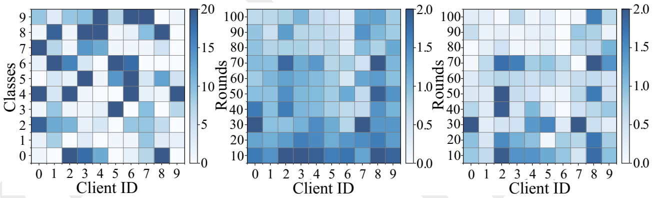

Figure 1: Illustration of the challenges associated with Non-IID data distribution among clients in FL. Heatmaps depict the distribution of 10 classes across 10 clients in the MelbournePedestrian dataset (*left*) and show the loss distribution over 100 training rounds for the FL process using the given Non-IID data partitioning with the SGD optimizer (*mid*) and the Adam optimizer (*right*), respectively.

Most research on backdoor defenses in FL has been primarily focused on server-side strategies [Rodríguez-Barroso *et al.*, 2023]. In these approaches, the server evaluates each client model, excluding any anomalies or selecting only trusted clients to participate in the aggregation process [Sattler *et al.*, 2020]. Many defense methods operate under the assumption of independent and identically distributed (IID) client data, relying on mean or median operations to construct global models [Yin *et al.*, 2018]. However, these methods lose their effectiveness when faced with Non-IID data, as illustrated in Figure 1 (*left*).

Although recent clustering methods [Wang *et al.*, 2023] enhance the efficacy of defenses, they struggle to address the non-stationary characteristics of client models. As demonstrated in Figure 1 (*mid*), in the FL attack environment, the objectives of each client are non-stationary, and their gradients exhibit noise across different FL iteration rounds. If the optimization method used, such as the commonly used stochastic gradient descent (SGD) [Wang *et al.*, 2023], cannot effectively tackle these challenges, the robustness and efficiency of backdoor defense will be significantly undermined.

To effectively identify and eliminate suspected attacking

[*] Corresponding Author

clients in the FL process, we propose a backdoor defense method, FLAC, which core idea is to combine coarse-grained and fine-grained clustering methods to identify and remove malicious models in a Non-IID setting. For the coarse-grained approach, we employ a minimum spanning tree (MST) to differentiate between benign clients and potential attackers. For fine-grained clustering, we use a density-based strategy that computes distances using weighted local model parameters and the direction of their updates. To address the effects of nonstationary objectives and noisy gradients from clients, we leverage the Adam optimizer [Kingma, 2014] for federated model optimization, providing proof of its consistent convergence under FL conditions. As indicated in Figure 1 (*right*), the loss distributions between different clients using Adam optimizer are more stable compared to those using SGD. The lighter blue colours in the subfigure also suggest that Adam converges more quickly than SGD. The main contributions of this paper can be summarized as follows:

- To the best of our knowledge, we are the first to present the theoretical convergence of the Adam optimizer in the context of attack and defense within FL.
- We propose a multi-granularity clustering that effectively isolates attackers, ensuring robust performance in FL with malicious participants and Non-IID data.
- We evaluate the performance of FLAC on both time series and image datasets, demonstrating that FLAC is highly effective in mitigating backdoor attacks.

2 Related Work

2.1 Backdoor Attack

FL attacks can be categorized into data poisoning and model poisoning based on the attacker's targets [Gu and Bai, 2023]. **Data Poisoning.** Biggio et al. [Biggio *et al.*, 2012] was the first to introduce data poisoning attacks in traditional machine learning. Sun et al. [Sun *et al.*, 2019] examined the impact of backdoor attacks on federated systems' performance, employing simple label flipping to create adversarial data [Tolpegin *et al.*, 2020; Wang *et al.*, 2020a]. Zhang et al. [Zhang *et al.*, 2019; Zhang *et al.*, 2020] utilized adversarial networks to facilitate data poisoning attacks. While the aforementioned data poisoning models primarily focus on images, Jiang et al. [Jiang *et al.*, 2023] proposed a generative approach for time series backdoor attacks. Chen et al. [Chen *et al.*, 2024] combined time series shapelets with differential evolution to introduce local perturbations in time series data. Additionally, Kasyap and Tripathy [Kasyap and Tripathy, 2024] generate poisoned samples using hyper-dimensional computing to induce misclassification.

Model Poisoning. In the domain of model poisoning, Bagdasaryan et al. [Bagdasaryan *et al.*, 2020] introduced a method that undermines the performance of the global model by submitting malicious updates to the server. Building on this foundation, Bhagoji et al. [Bhagoji *et al.*, 2019] developed a strategy that enhances the attack's efficacy through alternating optimization of stealth and antitargeting. Conversely, Wang et al. [Wang *et al.*, 2020a] executed a successful black-box attack on federated image classification, assuming the attacker has access to both the model's parameters and its structure. To improve attack effectiveness, Li et al. [Li *et al.*, 2023a] proposed an adaptive, scalable, multi-layer FL attack model capable of launching covert model poisoning attacks under black-box conditions. Additionally, some strategies enhance backdoor attacks by distributing the backdoor triggers. This involves dividing the original backdoor triggers into smaller segments and assigning these new triggers to different attackers, allowing each attacker to utilize a distinct small trigger for the backdoor attack [Lyu *et al.*, 2023]. Moreover, implementations of backdoor attacks have been explored using reinforcement learning and adversarial training techniques [Li *et al.*, 2023b].

2.2 Defense Method

Federated defense methods can be categorized into three major types: server-side, client-side, and communication channel defenses [Rodríguez-Barroso *et al.*, 2023].

In server-side defenses, a common approach is to eliminate anomalous client-side updates by employing anomaly detection techniques to identify and exclude suspicious model distributions from aggregation. For instance, Sattler et al. [Sattler *et al.*, 2020] utilize cosine distance to detect potential attack models, while Azulay et al. [Azulay *et al.*, 2020] employ leave-out methods for the same purpose. Additionally, robust aggregation methods have been developed to statistically select estimates that are more stable than anomalies or extreme values, thereby enhancing the reliability of global models [Blanchard *et al.*, 2017]. Recent research [Wang *et al.*, 2023][Nguyen *et al.*, 2021][Chen *et al.*, 2023] has also introduced clustering-based frameworks and dynamic clipping strategies to further improve defense effectiveness.

Client-side defenses configure algorithms locally, making them robust as they ensure clients are benign and provide individual protection for each participant [Portnoy *et al.*, 2022][Cao *et al.*, 2021]. In contrast to these strategies, communication channel-based defenses focus on safeguarding user privacy. These approaches enable multiple participants to collaboratively train a global model while protecting the exchange of model parameters and gradient updates from unauthorized access and eavesdropping [Gu *et al.*, 2024].

Despite their advantages, existing approaches fail to account for the non-stationary objectives of client models in the FBA setting. To address this issue, we introduce the Adam stochastic optimization method and demonstrate its theoretical convergence properties within the FL framework.

3 Preliminaries

The goal of FL is to minimize the average loss function $\mathcal{L}(w)$ of each participant's local model, defined as

$$\min_{w}\mathcal{L}\left(w\right)=\frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i\left(w\right)=\frac{1}{N}\sum_{i=1}^{N}\frac{1}{|\mathcal{D}_i|}\sum_{j=1}^{|\mathcal{D}_i|}l(X_i^j,y_i^j;w), \tag{1}$$

where N denotes the number of participants, w represents the global model parameters, and $\mathcal{D}_i$ indicates the private data of the i-th participant, such that $\mathcal{D}=\mathcal{D}_1\cup\ldots\cup\mathcal{D}_N$. The pair

(X_i^j, y_i^j) represents the j-th instance of the i-th client with label y_i^j, and $\mathcal{L}_i(w)$ is the loss function of the i-th participant.

In the t-th training round of FL, each participant receives a model w_g^t from the server, trains a local model using their data, and returns the updated model w_i^{t+1} to the server for aggregation into a new global model w_g^{t+1}. This updated model is then redistributed to participants for the next training round, continuing until convergence is achieved. Various aggregation strategies have been proposed for FL [Wang *et al.*, 2020b]. We employ FedAvg [McMahan *et al.*, 2017] to compute the average update value of the participating client set P for simplicity and efficiency.

$$w_g^{t+1} = w_g^t + \sum_{i \in P} \frac{|\mathcal{D}_i|}{\sum_{j=1}^{P} |\mathcal{D}_j|} (w_i^{t+1} - w_g^t). \tag{2}$$

Federated Backdoor Attack. FBA aims to poison the local data distribution $\mathcal{X}$ or manipulate model parameters in a way that adversely affects both benign models and the global model. To maintain stealthiness and robustness, FBA must balance the trade-off between the accuracy of each local model and the attack's success rate, formulated as

$$\begin{aligned} w_g^* &= \underset{w_g \in w}{\arg\min} \sum_{i \in \{C_b, C_p\}}^{N} \frac{|\mathcal{D}_i|}{\sum_{j=1}^{N} |\mathcal{D}_j|} \mathcal{L}_i(w_g) \\ &\triangleq E_{\mathcal{D}_p \sim \mathcal{X}_p} [P(\mathcal{D}_p; w_g) = \tau_p] + E_{\mathcal{D}_b \sim \mathcal{X}_b} [P(\mathcal{D}_b; w_g) = \tau_b]. \end{aligned} \tag{3}$$

Here, C_b denotes the set of benign clients, and C_p the malicious clients. The target τ corresponds to the desired outcome of the poisoned or original data during training, and $P(\cdot)$ is the inference function used to evaluate the global model.

4 Method

As shown in Figure 2, FLAC's architecture consists of three primary parts: similarity calculation, multi-granularity clustering, and adaptive clipping. Additionally, the framework's convergence properties and computational overhead (time and memory consumption) are also analyzed.

4.1 Similarity Calculation

In FL, especially in Non-IID settings, client gradient updates often exhibit significant variations in magnitude. Relying solely on traditional distance metrics, such as Mahalanobis or Euclidean distance, may fail to capture the crucial information about the alignment of gradient directions. To address this, we propose the use of the cosine distance, which simultaneously considers both the magnitude and direction of updates. By comparing the parameter updates w and their corresponding direction vectors v, this dual approach enables a more comprehensive assessment of horizontal parameter differences and vertical gradient directions.

In the model parameter weighing strategy, the parameters of the initial layers in the local model substantially influence feature extraction, whereas the output layer plays a pivotal role in determining classification performance. Given that attackers aim to manipulate classification outcomes, when concatenating model parameters w_i into a vector w_i', the output layer's weights should be assigned a higher value compared to other parameters (e.g., 1.5 times).

Attackers share backdoor objectives leading to similar gradients, while honest clients optimize for local performance, creating directional differences. We track model update directions by measuring parameter changes across consecutive rounds to identify malicious behavior.

The sum of the similarity between the weighted model parameters and the update directions is used to compute the distance matrix between client models. This is given by

$$H_{i,j} = \alpha(1 - \cos(w_i', w_j')) + (1 - \alpha)(1 - \cos(v_i, v_j)), \tag{4}$$

where w_i' represents the weighted vector of local model parameters for the i-th client, and $v_i = w_i^t - w_i^{t-3}$ denotes the update direction for the i-th client. The coefficient α lies within the interval (0, 1). This computation ensures that the elements of the matrix $H \in R^{p \times p}$ lie between 0 and 2, with smaller cosine distances indicating higher similarity.

4.2 Multi-Granularity Clustering

To effectively identify and eliminate suspected malicious clients in Non-IID scenarios, we employ both coarse-grained and fine-grained clustering methods to detect and remove compromised models. Traditional density-based clustering algorithms tend to fail when the proportion of attackers is high; thus, we introduce the MST method as a coarse-grained approach. This method identifies the edge with the highest anomalous similarity value, thereby locating regions with potentially compromised models. For the fine-grained clustering, DBSCAN [Schubert *et al.*, 2017] is employed, since the similarity distribution among clients is uneven and it can effectively identify boundaries in irregular cluster formations.

During the coarse-grained clustering process, Kruskal's algorithm [Kruskal, 1956] is employed to compute the MST, from which the largest edge is identified to partition the tree into two subtrees, resulting in two clusters, c_1 and c_2. According to the assumption in previous work [Wang *et al.*, 2023], the similarity among benign clients is lower than that among attackers. This occurs because attackers typically share a common objective, resulting in a higher degree of similarity between their models, which places them closer in terms of distance. Consequently, we derive the relationship $H_{\text{benign, poison}} > H_{\text{benign 1, benign 2}} > H_{\text{poison 1, poison 2}}$. In contrast, the intra-cluster distance variation is significant, and the cluster with tighter intra-cluster distances is flagged as potentially containing attackers. For instance, if $\mu_{c_1} < \mu_{c_2}$, then clients in cluster c_1 are likely to be identified as attackers.

For fine-grained clustering, the DBSCAN algorithm is employed, allowing nodes to self-organize into clusters while accommodating outliers. The clustering threshold is set at the 60th percentile of the similarity matrix H, and the resulting clusters are analyzed further. To handle cases where no attackers are present, the algorithm checks whether all participants cluster into a single group; if they do, it is inferred that no attackers are present. Otherwise, the participant with the highest variance within the cluster (e.g., c_3) is identified as an attacker. Importantly, if the identified attackers exceed 50% of the total clients, the clustering process is bypassed, as such scenarios are unrealistic.

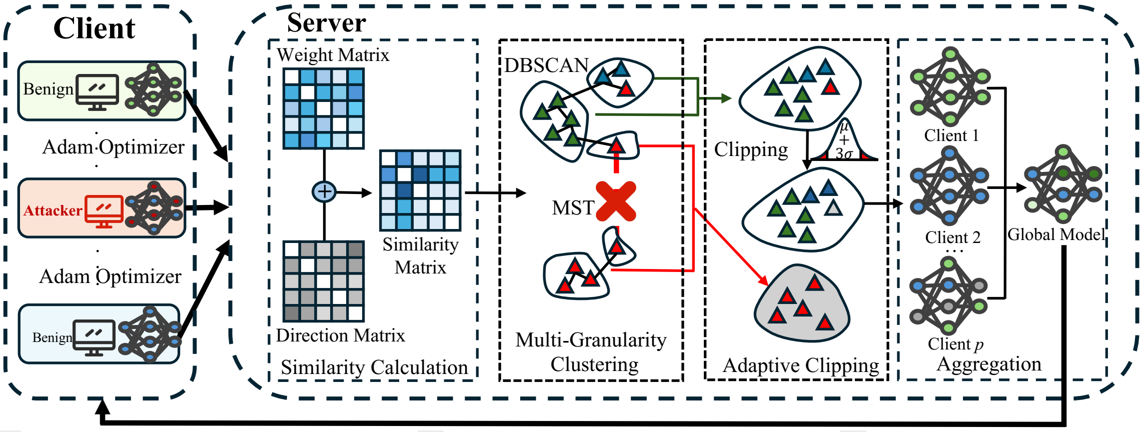

Figure 2: Overview of the FLAC algorithm.

Given that FBA does not manifest obvious anomalies in normal input data, it is both covert and poses a significant threat. Therefore, the combination of coarse- and fine-grained methods (i.e., $C_p = c_1 \cup c_3$, $C_b = [P]$ - C_p) is essential for ensuring robust detection and removal of malicious clients.

4.3 Adaptive Clipping

Some attackers may evade detection after the clustering step designed to eliminate suspicious clients. We employ adaptive clipping to mitigate the influence of remaining attackers on the global model. Previous works [Nguyen *et al.*, 2021; Wang *et al.*, 2023] have introduced clipping strategies, but these approaches often rely on fixed or heuristically determined thresholds. Selecting an appropriate clipping threshold remains a significant challenge.

During model training, the magnitude of updates from benign clients tends to decrease over time, due to constraints imposed by the optimization function [Nguyen *et al.*, 2021]. Unlike prior approaches, our method eliminates the majority of suspicious clients during the clustering stage, leaving only a small number of malicious clients to be handled by the clipping process. Building on these insights, we propose a parameter statistics-based adaptive clipping strategy within our FLAC method. The mean ($w_\mu = \frac{1}{p}\sum_{i=1}^{p} w_i$) and variance ($w_\sigma = \frac{1}{p}\sum_{i=1}^{p}(w_i - w_\mu)^2$) of model parameters from clients in the benign cluster are first calculated. Then the sum of w_μ and $3\sqrt{w_\sigma}$ forms the potential intermediate model $w_{\text{mid}} = w_\mu + 3\sqrt{w_\sigma}$.

Finally, the L_2-norm of w_{mid} is used as the clipping threshold. During aggregation, the clipping strategy from Eq. 5 is applied to mitigate outlier impacts:

$$w_g^{t+1} = \frac{1}{|C_p|}\sum_{i \in C_p} w_i^t \cdot min(1, \frac{\rho_t}{||w_i^t||}). \tag{5}$$

Here, $\rho_t = \|w_{\text{mid}}^t\|$ is the clipping threshold for the t-th round.

4.4 Time and Memory Consumption Analysis

The time complexity of the training of each client is $\mathcal{O}(|\mathcal{D}| \cdot R_{train} \cdot N_{\text{param}})$, where N_{param} represents the number of parameters in the model, R_{train} the training rounds. For our server-side defense FLAC, its time complexity is $\mathcal{O}(N^2 \cdot N_{\text{param}})$. Since $|\mathcal{D}| \cdot R_{train} \gg N^2$ in our setting, the computational time of FLAC is negligible. A detailed analysis of the time complexities can be found in Appendix C.

Regarding memory consumption, while Adam theoretically requires tripled memory compared to SGD due to storing two additional values per parameter [Shazeer and Stern, 2018], our empirical measurements show that the actual overhead ranges from 1.01× to 1.68× depending on model architecture, as shown in Appendix A2, which becomes negligible as communication bandwidth and computational costs dominate resource considerations by orders of magnitude.

4.5 Convergence Analysis

In our defense, each selected client employs the Adam optimizer for model updates, as described in Eq. 6:

$$w_i^{t+1} = w_g^t - \eta\frac{m_t}{\sqrt{\hat{v}_t}+\epsilon}, m_t = (1-\kappa_1)\sum_{i=1}^{t}\kappa_1^{t-i}\nabla\mathcal{L}_i(w_g^t),$$
$$\hat{v}_t = (1-\kappa_2)diag(\sum_{i=1}^{t}\kappa_2^{t-i}\nabla\mathcal{L}_i(w_g^t)^2), \tag{6}$$

where η is the learning rate, m_t represents the biased first moment estimate, and $\hat{v}_t$ is the biased second raw moment estimate. The parameters κ_1 and κ_2 are the coefficients used for updating these moment estimates. Compared to the convergence analysis of SGD in FL, and following the conclusions of [Reddi *et al.*, 2019], the key challenge for Adam lies in its denominator, which represents the biased second raw moment estimate. Establishing a lower bound for this term is critical. To address this, we adopt the proof concepts from [Chen *et al.*, 2023], making several necessary assumptions. Theorem 1 outlines the resulting convergence analysis.

Assumption 1: For the expected loss function $\mathcal{L}(w)$ and any possible model parameter w, $\hat{w}$, $\mathcal{L}(w)$ is said to be β-smooth if it satisfies

$$||\nabla\mathcal{L}(w) - \nabla\mathcal{L}(\hat{w})|| \leq \beta||w - \hat{w}||, \tag{7}$$

where $||\cdot||$ denotes the L_2-norm.

Assumption 2: For all w, let z_i^j be sampled uniformly at random from the local data $\mathcal{D}_i$ of the i-th client, assuming that the variance of the stochastic gradient $\nabla\mathcal{L}_i(w; z_i^j)$ in each client has an upper bound B^2, i.e., it satisfies

$$E||\nabla\mathcal{L}_i(w; z_i^j) - \nabla\mathcal{L}_i(\overline{w})||^2 \leq B^2. \tag{8}$$

Assumption 3: In FL, both malicious and benign clients optimize their respective objectives independently. Therefore, the expectation-squared paradigm of the stochastic gradient is uniformly bounded. For all w, there exists an upper bound G on the gradient $\nabla\mathcal{L}_i(w)$.

$$E||\nabla\mathcal{L}_i(w)||^2 \leq G^2. \tag{9}$$

Assumption 4: For all w, it is assumed that the diagonal elements of the squared stochastic gradient $\nabla\mathcal{L}_i(w)$ in each client have a uniform lower bound, i.e., they satisfy the following Eq.10 .

$$diag(\nabla\mathcal{L}_i(w)^2) \geq \mathit{\Gamma}^2. \tag{10}$$

Lemma 1: For $\forall i, t, \varphi_i^t = min\{1, \frac{\rho_t}{||\Delta w_i^t||}\}$, where $i \in [p_t]$ denotes the participant indexes in the aggregation rounds, and $t \in [T]$ denotes the rounds. After clustering and removing suspicious models such that $p_t \geq p/2$, and assuming the clipping threshold ρ_t is greater than the smallest local model paradigm, i.e., $\rho_t \geq min\{||\Delta w_i^t|||\}$ and $1 - \phi_t/p_t \leq Pr_{i\in[p_t]}(||\Delta w_i^t|| \leq \rho_t) \leq 1$. Then we have $1 - \phi_t/p_t \leq E_{i\in[p_t]}\varphi_i^t \leq 1$, where ϕ_t represents the number of attackers that evade the defense method at round t.

Theorem 1: According to the above Assumptions 1-4. after training T rounds, FLAC can converge to the global optimum w_g^*. Therefore, the convergence described by Eq.11 can be obtained.

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T} E||\nabla\mathcal{L}(w_g^{t-1})||^2 &\leq \frac{4}{T}E[\mathcal{L}(w_g^1) - \mathcal{L}(w_g^*)] + \frac{2\beta}{T}\Omega \\ &+\frac{8\eta^2B^2\mathcal{M}^2}{p} + \frac{4+16\eta^2\mathcal{M}^2-16\eta\mathcal{M}}{p^2} + \frac{32\mathcal{M}\eta^2B^2\Phi}{p^3},\end{aligned} \tag{11}$$

where $\Omega = \sum_{t=1}^{T}\rho_t^2$, $\mathcal{M} = (1-\kappa_1^t)/\sqrt{1-\kappa_2^t}\Gamma$ and $\Phi = \sum_{t=1}^{T}\phi_t$. The proof procedure of Lemma 1 and Theorem 1 are in Appendix A.

5 Experiments

In this section, we begin by detailing the experimental setup and demonstrate the effectiveness of our proposed FLAC method[1] by comparing it with classical defense approaches under various scenarios. Additionally, we assess the contribution of each individual module within our framework and provide an in-depth analysis of the method's performance through case studies.

5.1 Experimental Setup

We conduct our experiments on both time series and image datasets within a simulated FL environment, implemented in PyTorch, with computations performed using CUDA 11.2.

We set 300 aggregation rounds as the convergence benchmark. As shown in Fig. 3, under non-attacking scenarios, all defense algorithms reach convergence by this point, ensuring a fair and consistent evaluation of their performance. Defense mechanisms are applied at each round to simulate realistic federated learning scenarios. Furthermore, to simulate Non-IID data distributions, the training data for each client was partitioned using a Dirichlet distribution, which allows for varying levels of data heterogeneity across clients.

Datasets. For the time series datasets, we adopt the settings from FLATS [Chen *et al.*, 2024]. For image datasets, we evaluate our method on three common benchmark datasets: CIFAR-10 [Krizhevsky *et al.*, 2009], MNIST [LeCun *et al.*, 1998], and FMNIST [Xiao *et al.*, 2017]. Additionally, we employ FCN [Yang *et al.*, 2022] as the training model for time series datasets, while LeNet [LeCun *et al.*, 1998] is utilized for image datasets. More dataset information can be found in Appendix D.

Baselines. We evaluate the effectiveness of our defense method by comparing it against five state-of-the-art defense methods: NDC [Sun *et al.*, 2019], DP [Geyer *et al.*, 2017], Krum [Blanchard *et al.*, 2017], FoolsGold [Fung *et al.*, 2018], and RFA [Pillutla *et al.*, 2022].

For the FBA methods, we utilize FLATS [Chen *et al.*, 2024] and its variants for time series datasets. For image datasets, we assess the performance of various defense methods under attacks from LF [Jebreel *et al.*, 2023], MP [Bagdasaryan *et al.*, 2020], and 3DFed [Li *et al.*, 2023a].

Evaluation Metric. To assess the performance of our method and the baseline defenses, we measure the classification accuracy (ACC) of the aggregated model on the test dataset. Compared to the original method without defense (or under attack), the smaller the ACC drop, the more effective the defense method.

$$ACC(\mathcal{D}_{test}, f) = \frac{1}{|\mathcal{D}_{test}|}\sum_{X_i\in\mathcal{D}_{test}} \mathbf{1}(f(X_i) = y_i). \tag{12}$$

5.2 Experimental Results

In this section, we provide detailed experimental evidence to support the effectiveness of our defense method, FLAC.

No Attacker Case. Robust federated algorithms must maintain effectiveness without attackers. Figure 3 compares the performance of 6 defense algorithms on the MelbournePedestrian and ElectricDevices datasets. Both FLAC and RFA achieve ACC values comparable to FedAvg, with FLAC showing a more stable performance curve. In contrast, DP, NDC, and FoolsGold yield slightly lower ACC, while Krum's performance is significantly reduced due to its exclusion of many clients to mitigate attacker influence. These results indicate that FLAC maintains global model convergence, aligning with the convergence analysis.

Time Series Attack Table 1 presents a comparison of FLAC's performance against other defense baselines when

[1] The source code for FLAC is available at https://github.com/catb62/FLAC

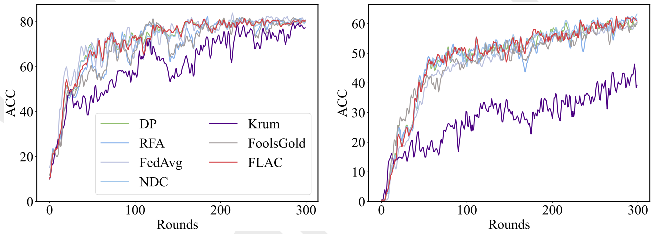

Figure 3: Performance of defense algorithms without attackers on MelbournePedestrian dataset (*left*) and ElectricDevices (*right*)

the federated system is under attack by FLATS. The symbol "↓" denotes a drop in ACC after applying defense mechanisms. As shown in bold in Table 1, FLAC achieves the best defense performance across 7 out of 8 datasets. Notably, on the TwoPatterns dataset, the ACC increased from 83.19% w/o Attack to 84.33% after implementing FLAC, further demonstrating its superior ability to mitigate the attacker's influence compared to other methods. The presence of "↓" for both NDC and DP defense algorithms suggests that these approaches may hinder the contributions of benign clients to the global model. FLAC reached only 87.99% accuracy on the ECG5000 dataset, while NDC and RFA outperformed in this instance without an attack, likely because FLAC's clipping mechanism removed contributions from normal clients.

Dataset	FLAC	NDC	DP	Krum	FoolsGold	RFA	w/o Def	w/o Attack
ECG5000	87.99	**92.01**	59.92	89.96	90.43	91.41	60.49	91.07
ElectricDevices	**62.78**	62.33	61.88	62.13	62.07	59.77	52.32	63.25
FaceAll	**56.73**	50.89	38.18 ↓	53.64	39.74	55.36	47.23	89.40
MelbournePed	**74.11**	63.08	58.17	63.08	59.83	58.44	47.98	81.22
ShapesAll	**12.13**	6.67	1.84 ↓	7.03	8.11	10.78	6.14	16.04
SwedishLeaf	**61.82**	60.44	56.65	61.32	59.29	56.76	34.38	87.78
TwoPatterns	**84.33**	62.42 ↓	67.31	74.44	72.38	79.79	63.53	83.19
UWaveGestureLX	**65.28**	52.45	46.75	48.24	50.29	56.67	45.24	72.01

Table 1: Comparison of FLAC with other baselines (%)

Figure 4 (*left*) illustrates the performance comparison of 6 defense algorithms on the FaceAll dataset. FLAC demonstrates greater stability with less fluctuation in ACC (red line), achieving an average value of 56.73%. In contrast, both the DP and FoolsGold algorithms exhibit limited performance during the first 200 iterations in the absence of attackers. Moreover, the NDC algorithm shows a continuous decline in ACC during the last 100 iterations when under attack, and the RFA algorithm results in significant fluctuations in ACC, indicating a lack of stability.

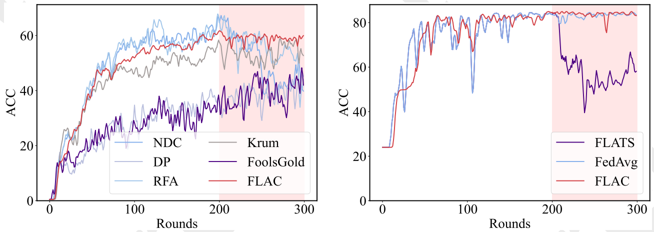

Figure 4: Comparison of 6 defense methods on FaceAll dataset (*left*) and the attacks and defenses on TwoPatterns dataset (*right*).

Additionally, we conducted defense experiments against various attack methods for the time series classification task in a FL scenario. The results, as presented in Table 2, demonstrate that FLAC effectively defends against all 4 different attacks. Notably, the "↑" symbol indicates an upward trend in ACC after applying the defense (w/ Def). The model's performance, particularly on the ElectricDevices and TwoPatterns datasets, shows minimal deviation compared to the w/o Attack scenarios, further underscoring the robustness and effectiveness of FLAC in mitigating the impact of these attacks.

Dataset	FLATS		AttackRandShape		AttackRandAll		AttackOnePoint	
	w/o Def	w/ Def	w/o Def	w/ Def	w/o Def	w/ Def	w/o Def	w/ Def
ECG5000	60.49	87.99 ↑	88.96	90.56 ↑	86.82	92.29 ↑	84.86	92.31 ↑
ElectricDevices	53.32	62.78 ↑	61.67	62.68 ↑	61.38	61.70 ↑	62.25	63.19 ↑
FaceAll	47.23	56.73 ↑	67.41	80.60 ↑	75.97	81.50 ↑	71.61	83.14 ↑
MelbournePedestrian	47.98	74.11 ↑	52.81	76.64 ↑	49.04	77.52 ↑	58.97	77.30 ↑
ShapesAll	6.14	12.13 ↑	8.11	11.80 ↑	6.44	7.88 ↑	12.37	16.83 ↑
SwedishLeaf	31.19	61.82 ↑	35.40	60.95 ↑	38.79	59.91 ↑	32.27	61.79 ↑
TwoPatterns	63.53	84.33 ↑	55.22	83.79 ↑	55.51	83.80 ↑	67.63	83.78 ↑
UWaveGestureLibraryX	45.24	65.28 ↑	49.70	71.33 ↑	58.99	69.74 ↑	68.51	69.29 ↑

Table 2: FLAC performance under 4 time series attack methods (%)

Figure 4 (*right*) illustrates the ACC curves for three scenarios on the TwoPatterns dataset: without attack (FedAvg), with attack (FLATS), and with both attack and defense (FLAC). The purple line indicates a sharp decline in ACC due to the FLATS attack, while the red line demonstrates that FLAC significantly restores ACC to levels comparable to the attack-free scenario, highlighting its effectiveness in enhancing system reliability under attack. Furthermore, in the 0–200 round range, where no attack occurs, FLAC maintains convergence and performance akin to the no-attack scenario, suggesting that its defense mechanism minimally impacts benign clients and preserves robust performance in the absence of attacks.

Attack on Images To further validate the effectiveness of the proposed FLAC, we conduct experiments on 3 image datasets using 3 FBA algorithms and 6 defense methods. As shown in Table 3, FLAC achieves the best results in 8 out of 9 tests. Under various FBA scenarios (i.e., LF, MP, and 3DFed), FLAC attains an accuracy closer to the w/o Attack setting and shows a larger improvement compared to the w/o Def situation.

Method	Dataset	FLAC	NDC	DP	Krum	FoolsGold	RFA	w/o Def	w/o Attack
LF	CIFAR-10	**88.43**	84.84	73.29	75.34	84.65	85.10	44.19	89.20
	MNIST	**98.24**	95.56	84.47	89.21	96.17	96.83	69.17	99.16
	FMNIST	**81.94**	77.53	77.69	68.44	79.08	56.53 ↓	65.39	87.79
MP	CIFAR-10	**85.42**	81.08	79.63	81.37	84.76	84.98	44.95	89.20
	MNIST	95.97	93.66	78.32	88.20	**96.81**	94.75	64.18	99.16
	FMNIST	**86.58**	68.18	85.77	79.99	86.45	85.30	68.17	87.79
3DFed	CIFAR-10	**87.62**	84.10	77.97	81.45	86.76	86.35	43.54	89.20
	MNIST	**98.92**	97.45	88.32	81.03	98.17	98.11	68.13	99.16
	FMNIST	**78.00**	74.65	76.09	65.64 ↓	77.60	43.40 ↓	68.29	87.79

Table 3: FLAC performance on image datasets (%)

5.3 Ablation Experiment

This subsection presents ablation studies to verify the necessity of each component in the FLAC method. First, we assess the importance of parameter weighting and model update direction, denoted as FLAC-NW and FLAC-NV, respectively. Next, we evaluate FLAC-NT, which omits the MST algorithm, and FLAC-ND, which excludes the DBSCAN method. Additionally, we investigate the impact of clipping by introducing FLAC-NC, which removes the clipping component.

Lastly, to validate the effectiveness of the Adam optimizer, we replace it with SGD, referring to FLAC-NA.

Dataset	FLAC	FLAC-NC	FLAC-ND	FLAC-NT	FLAC-NV	FLAC-NW	FLAC-NA
ECG5000	87.99	**92.38**	77.04	78.33	87.79	87.01	60.76
ElectricDevices	**62.78**	55.43	45.77	43.90	59.65	57.23	47.30
FaceAll	**56.73**	55.15	44.66	16.54	54.25	53.42	27.87
MelbournePed	**74.11**	72.12	65.17	28.62	64.25	66.64	61.17
ShapesAll	**12.13**	10.78	10.99	8.70	9.53	10.64	9.50
SwedishLeaf	**61.82**	61.82	55.81	45.59	60.45	59.11	50.56
TwoPatterns	**84.33**	81.49	41.28	45.22	69.02	66.38	74.67
UWaveGestureLX	**65.28**	62.91	61.73	55.89	63.02	62.22	41.76
Average	**63.15**	61.51	50.31	40.35	58.50	57.83	46.70
Difference	/	1.64	12.84	22.80	4.65	5.32	16.45

Table 4: FLAC ablation study under 8 dataset (%)

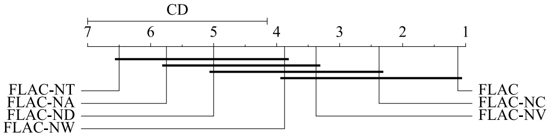

Figure 5: CD plot for the FLAC and its variants.

Table 4 shows FLAC outperforms the second-best FLAC-NC by 1.64. Removing parameter weighting or update direction (FLAC-NW, FLAC-NV) results in worse defense, highlighting the importance of similarity calculation. Omit clustering (FLAC-NT, FLAC-ND) or replace Adam with SGD (FLAC-NA) leads to the poorest defense, stressing the role of clustering and Adam in identifying attackers.

Figure 5 presents the critical difference (CD) plot for the 7 evaluated strategies. FLAC ranks highest, followed closely by FLAC-NC, indicating effective identification and removal of most attackers during the multi-granularity clustering phase. The CD plot further reveals that FLAC significantly outperforms FLAC-NT, FLAC-NA, and FLAC-ND.

Figure 6 compares the performance of the Adam and SGD optimizers on the FaceAll dataset. In terms of ACC, the Adam optimizer (blue line) reaches higher levels in fewer rounds compared to the SGD (red line). Similarly, for loss, the Adam optimizer achieves lower values more quickly than SGD. These results demonstrate that the FLAC is more efficient and stable with the Adam optimizer in FL scenarios.

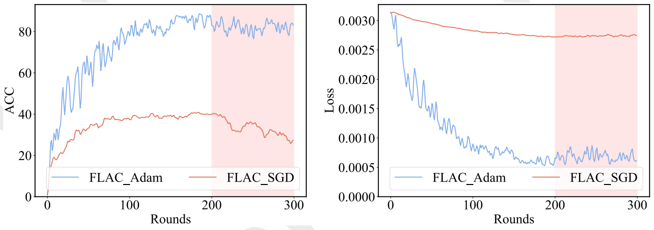

Figure 6: Illustration of the comparison of Adam and SGD regarding ACC (*left*) and Loss (*right*) on the Face All dataset. The red-shaded area indicates the attack-defense interval.

5.4 Case Study

To emphasize the role of clustering in the defense process, Figure 7 illustrates the cumulative number of attacker appearances (Attacker Number), the cumulative count of successfully identified attackers through our multi-granularity clustering (Identified Attackers), and the cumulative total of rejected clients (Rejected) over 300 communication rounds. The results of eight time series datasets demonstrate that the FLAC algorithm effectively rejects more than 90% of attackers while maintaining strong overall performance. This highlights the critical role of coarse and fine-grained clustering in identifying malicious participants, with few evading attackers emphasizing the necessity of clipping operations.

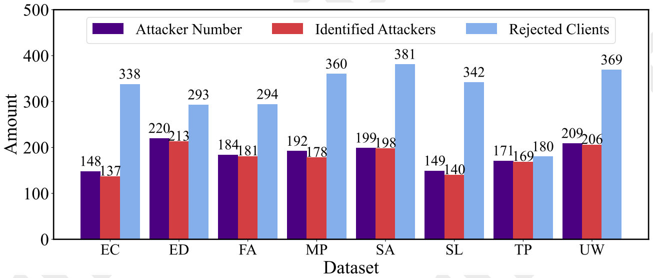

Figure 7: Cumulative number of attackers identified by FLAC across all training rounds. Dataset abbreviations: EC (ECG5000), ED (ElectricDevices), FA (FaceAll), MP (MelbournePedestrian), SA (ShapesAll), SL (SwedishLeaf), TP (TwoPatterns), UW (UWaveGestureLibraryX).

Furthermore, Figure 8 illustrates the complementary roles of MST and DBSCAN on the ElectricDevices dataset. The highlighted sections show the distinct contributions of each method, where MST identifies 170 attackers and DBSCAN identifies 165 ones. The overlapping areas represent the 122 attackers detected by both methods. This combination provides a robust and comprehensive elimination of suspicious clients, underscoring the importance of integrating both methods within the FLAC. As a further step, additional experiments can be found in Appendix E, which assess the impact of parameter selection and evaluate the robustness of FLAC against varying attacker proportions.

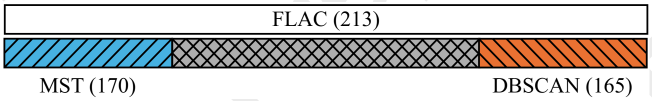

Figure 8: The complementary roles of MST and DBSCAN.

6 Conclusion

This paper proposes a robust federated backdoor defense algorithm leveraging Adam optimizer and multi-granularity clustering to accelerate learning and attacker detection. The approach integrates model parameter update magnitudes and gradient update directions for dual-level clustering analysis, enhanced by statistical pruning mechanisms to mitigate undetected adversarial influence. Theoretical analysis confirms convergence, while extensive experiments on time series and image datasets demonstrate stable defense performance. The future would combine server-side and client-side mechanisms for a more comprehensive defense system.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (No. 2024JBMC033), the National Natural Science Foundation of China (No. 62402031), and the Beijing Nova Program (No. 20240484620).

References

[Azulay *et al.*, 2020] Shahar Azulay, Lior Raz, Amir Globerson, Tomer Koren, and Yehuda Afek. Holdout sgd: Byzantine tolerant federated learning. *arXiv preprint arXiv: 2008.04612*, 2020.

[Bagdasaryan *et al.*, 2020] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR, 2020.

[Bhagoji *et al.*, 2019] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643. PMLR, 2019.

[Biggio *et al.*, 2012] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[Blanchard *et al.*, 2017] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30, 2017.

[Cao *et al.*, 2021] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Data poisoning attacks to local differential privacy protocols. In *30th USENIX Security Symposium*, pages 947–964. USENIX Association, 2021.

[Chen *et al.*, 2023] Zekai Chen, Shengxing Yu, Mingyuan Fan, Ximeng Liu, and Robert H Deng. Privacy-enhancing and robust backdoor defense for federated learning on heterogeneous data. *IEEE Transactions on Information Forensics and Security*, 2023.

[Chen *et al.*, 2024] Shengbo Chen, Jidong Yuan, Zhihai Wang, and Yongqi Sun. Local perturbation-based blackbox federated learning attack for time series classification. *Future Generation Computer Systems*, 158:488–500, 2024.

[Fung *et al.*, 2018] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866*, 2018.

[Geyer *et al.*, 2017] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[Gu and Bai, 2023] Yu-Hao Gu and Yue-Bin Bai. Survey on security and privacy of federated learning models. *Ruan Jian Xue Bao/Journal of Software (in Chinese)*, 34(6):2833–2864, 2023.

[Gu *et al.*, 2024] Hongyan Gu, Xinyi Zhang, Jiang Li, Hui Wei, Baiqi Li, and Xinli Huang. Federated learning vulnerabilities: Privacy attacks with denoising diffusion probabilistic models. In *Proceedings of the ACM on Web Conference 2024*, pages 1149–1157, 2024.

[Jebreel *et al.*, 2023] Najeeb Moharram Jebreel, Josep Domingo-Ferrer, David Sánchez, and Alberto Blanco-Justicia. Lfighter: Defending against the label-flipping attack in federated learning. *Neural Networks*, 170:111–126, 2023.

[Jiang *et al.*, 2023] Yujing Jiang, Xingjun Ma, Sarah Monazam Erfani, and James Bailey. Backdoor attacks on time series: A generative approach. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 392–403. IEEE, 2023.

[Kasyap and Tripathy, 2024] Harsh Kasyap and Somanath Tripathy. Beyond data poisoning in federated learning. *Expert Systems with Applications*, 235:121192, 2024.

[Kingma, 2014] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[Kruskal, 1956] Joseph B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Li *et al.*, 2022] Chengxi Li, Gang Li, and Pramod K Varshney. Federated learning with soft clustering. *IEEE Internet of Things Journal*, 9(10):7773–7782, 2022.

[Li *et al.*, 2023a] Haoyang Li, Qingqing Ye, Haibo Hu, Jin Li, Leixia Wang, Chengfang Fang, and Jie Shi. 3dfed: Adaptive and extensible framework for covert backdoor attack in federated learning. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1893–1907. IEEE, 2023.

[Li *et al.*, 2023b] Henger Li, Chen Wu, Senchun Zhu, and Zizhan Zheng. Learning to backdoor federated learning. *arXiv preprint arXiv:2303.03320*, 2023.

[Lyu *et al.*, 2023] Xiaoting Lyu, Yufei Han, Wei Wang, Jingkai Liu, Bin Wang, Jiqiang Liu, and Xiangliang Zhang. Poisoning with cerberus: Stealthy and colluded backdoor attack against federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9020–9028, 2023.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[Nguyen *et al.*, 2021] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Shaza Zeitouni, et al. Flame: Taming backdoors in federated learning (extended version 1). *arXiv preprint arXiv: 2101.02281*, 2021.

[Nguyen *et al.*, 2022] Dinh C Nguyen, Quoc-Viet Pham, Pubudu N Pathirana, Ming Ding, Aruna Seneviratne, Zihuai Lin, Octavia Dobre, and Won-Joo Hwang. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (Csur)*, 55(3):1–37, 2022.

[Pillutla *et al.*, 2022] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *IEEE Transactions on Signal Processing*, 70:1142–1154, 2022.

[Portnoy *et al.*, 2022] Amit Portnoy, Yoav Tirosh, and Danny Hendler. Towards federated learning with byzantine-robust client weighting. *Applied Sciences*, 12(17), 2022.

[Reddi *et al.*, 2019] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv: 1904.09237*, 2019.

[Rodríguez-Barroso *et al.*, 2023] Nuria Rodríguez-Barroso, Daniel Jiménez-López, M. Victoria Luzón, Francisco Herrera, and Eugenio Martínez-Cámara. Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges. *Information Fusion*, 90:148–173, 2023.

[Sattler *et al.*, 2020] Felix Sattler, Klaus-Robert Müller, Thomas Wiegand, and Wojciech Samek. On the byzantine robustness of clustered federated learning. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8861–8865, 2020.

[Schubert *et al.*, 2017] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017.

[Shazeer and Stern, 2018] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR, 10–15 Jul 2018.

[Sun *et al.*, 2019] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[Tolpegin *et al.*, 2020] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*, pages 480–501. Springer, 2020.

[Wang *et al.*, 2020a] 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. *Advances in Neural Information Processing Systems*, 33:16070–16084, 2020.

[Wang *et al.*, 2020b] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

[Wang *et al.*, 2023] Yongkang Wang, Di-Hua Zhai, Yongping He, and Yuanqing Xia. An adaptive robust defending algorithm against backdoor attacks in federated learning. *Future Generation Computer Systems*, 143:118–131, 2023.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[Yang *et al.*, 2022] Wenbo Yang, Jidong Yuan, Xiaokang Wang, and Peixiang Zhao. Tsadv: Black-box adversarial attack on time series with local perturbations. *Engineering Applications of Artificial Intelligence*, 114:105218, 2022.

[Yin *et al.*, 2018] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pages 5650–5659. PMLR, 2018.

[Zhang *et al.*, 2019] Jiale Zhang, Junjun Chen, Di Wu, Bing Chen, and Shui Yu. Poisoning attack in federated learning using generative adversarial nets. In *2019 18th IEEE international conference on trust, security and privacy in computing and communications/13th IEEE international conference on big data science and engineering (TrustCom/BigDataSE)*, pages 374–380. IEEE, 2019.

[Zhang *et al.*, 2020] Jiale Zhang, Bing Chen, Xiang Cheng, Huynh Thi Thanh Binh, and Shui Yu. Poisongan: Generative poisoning attacks against federated learning in edge computing systems. *IEEE Internet of Things Journal*, 8(5):3310–3322, 2020.