Improving Defenses against Backdoors in Federated Learning using Data Generation

IN5000: Final Project

Sebastiaan van Moergestel

Delft University of Technology

TU Delft

Improving Defenses against Backdoors in Federated Learning using Data Generation

by

Sebastiaan van Moergestel

to obtain the degree of **Master of Science**

in **Embedded Systems**

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)

at the Delft University of Technology

to be defended publicly on Wednesday March 12, 2025 at 17:00 PM.

Student number:	5421497
Project Duration:	Oktober, 2023 - March, 2025
Thesis committee:	Dr. Kaitai Liang (responsible supervisor)
	Rui Wang (co-supervisor)
	Prof. George Smaragdakis (the chair)
	Prof. Jérémie Decouchant

TU Delft

Summary

In this work, we propose a general solution to address the non-IID challenges that hinder many defense methods against backdoor attacks in federated learning. Backdoor attacks involve malicious clients attempting to poison the global model. While many defense methods effectively filter out these malicious clients using clustering techniques, their effectiveness diminishes when the federated learning process involves non-IID datasets. In such cases, clustering methods struggle to distinguish between benign and malicious clients due to the inherent variability in the clients' data distributions.

Our proposed solution leverages data generation to mitigate the non-IID nature of clients' local datasets. By generating synthetic data, the datasets become more IID, enabling defense methods to once again effectively counter backdoor attacks. Evaluations are carried out on standard datasets in the image classification fields, like MNIST and CIFAR-10. The results show that the data generation solution can effectively improve the performance of defense methods and filter out malicious clients again. Although the generated data samples may suffer from low quality and limited diversity due to constraints in training the generative adversarial networks (GANs), our approach demonstrates significant improvements in defending against backdoors.

Contents

Summary **i**

1 Introduction **1**
1.1 Problem Statement . . . 1
1.2 Thesis Object . . . 2
1.3 Contributions . . . 2
1.4 Thesis Outline . . . 2

2 Background Theory **3**
2.1 Federated Learning . . . 3
2.2 Backdoor Attacks . . . 4
2.2.1 Label Flipping . . . 4
2.2.2 Trojan-based Attacks . . . 4
2.3 Defensive Schemes . . . 5
2.3.1 Clipping . . . 5
2.3.2 Adding Noise . . . 5
2.3.3 Krum . . . 6
2.3.4 Trimmed-Mean . . . 7
2.3.5 FLAME . . . 7
2.4 Data Generation . . . 8
2.4.1 Synthetic Data Generation . . . 8

3 Problem Formulation **9**
3.1 Defense schemes on non-IID data . . . 9
3.2 Federated Learning Setup . . . 9
3.2.1 LeNet Architecture . . . 10
3.2.2 ResNet-18 Architecture . . . 10
3.3 Threat Model . . . 10
3.3.1 Backdoor attack . . . 10

4 Data Generation Solution **11**
4.1 Data Generation Techniques . . . 11
4.2 Generative Adversarial Networks . . . 11
4.3 Challenges and Assumptions . . . 12
4.3.1 Hyperparameters . . . 12
4.4 Framework and Algorithms . . . 12
4.4.1 GAN-based Data Generation . . . 12
4.4.2 GANs in Federated Learning . . . 12

5 Experiments **15**
5.1 Experimental Setup . . . 15
5.1.1 FL Configuration . . . 15
5.1.2 Datasets . . . 15
5.1.3 Byzantine Attacks . . . 15
5.1.4 Non-IID . . . 15
5.1.5 GAN . . . 17
5.1.6 Defense methods . . . 17
5.1.7 Evaluation . . . 17
5.2 Experimental Results . . . 17
5.2.1 Improving Defense Methods against Backdoor Attacks . . . 17
5.2.2 Quality of Generated Samples . . . 18

5.2.3 Mitigating Untargeted Backdoor Attacks 20
5.2.4 Summary 20

6 Conclusion 25

References 26

1 Introduction

Building machine learning (ML) models often require collecting massive amounts of training data from diverse sources. Traditional machine learning paradigms typically rely on centralized data collection and processing. However, this presents significant challenges related to data privacy, security, and regulatory compliance. Another problem is that in many industries, data is often dispersed and locked in multiple organizations, where data sharing is strictly forbidden due to the growing concerns about data privacy. Federated learning (FL) emerges as a novel distributed machine learning approach that addresses these challenges by enabling model training across multiple decentralized devices or servers while keeping the data localized [26].

Federated learning operates on the principle of decentralizing the training process, where local models are trained on devices using local data. These models are then aggregated into a global model by a centralized server without the need to transfer raw data, thereby preserving data privacy. FL not only enhances privacy but also reduces the risk of data breaches and adheres to data protection regulations such as the General Data Protection Regulation (GDPR) in Europe [5].

The concept of FL has found applications across various domains such as image recognition [16], natural language processing [7], healthcare [35] and finance [43] where sensitive data is important. For example, in healthcare, FL facilitates the development of predictive models for disease detection without compromising patient confidentiality [35].

Despite the promising advantages, FL also presents several research challenges. Model accuracy, communication efficiency, system heterogeneity, and security vulnerabilities need to be meticulously addressed to harness the full potential of FL [23].

1.1. Problem Statement

One of the most significant challenges facing federated learning systems is their vulnerability to backdoor and poisoning attacks [11]. In backdoor attacks, malicious clients that participate intentionally introduce harmful patterns into the training data or the model updates of the global model. These patterns are designed to cause the global model to misbehave in specific ways when encountering certain triggers. For example, an attacker could manipulate a model to misclassify images containing specific features, which can be particularly dangerous in applications such as autonomous driving or medical diagnosis [42]. Because of the decentralized nature of FL, the central server lacks direct access to the raw training data of the participating clients, making it challenging to detect and mitigate such malicious modifications.

Poisoning attacks are closely related to backdoor attacks, which involve injecting false data into the training process to decrease the overall model performance. This is done by corrupting the data held by participating clients or by altering the updates sent to the global model. FL relies on the contributions of multiple devices, even a small fraction of compromised devices can have a large impact on the global model's integrity.

Multiple approaches have been proposed to defend FL against these backdoor attacks. Some of the prominent defenses include anomaly detection techniques, robust aggregation methods, and differential privacy [38].

Despite these advancements, defending against backdoor attacks becomes significantly more challenging when dealing with non-IID datasets. In federated learning, non-IID data refers to the situation where the data distribution across participating clients is not uniform or identical, which is often the case in real-world applications [23]. Non-IID data can exacerbate the difficulty of distinguishing between benign and malicious updates because the natural variability in the data can resemble the patterns introduced by an attacker. This variability can undermine the effectiveness of anomaly detection and robust aggregation techniques, leading to higher false positive or false negative rates [47].

1.2. Thesis Object

The primary objective of this thesis is to improve the performance of existing defense methods on non-IID datasets. Specifically, this thesis aims to improve existing defense mechanisms through the application of data generation techniques to reduce the non-IID nature of the client's datasets.

1.3. Contributions

The contributions of this thesis are outlined as follows:

- Analysis of Current Defense Methods on Non-IID Datasets: This work begins with a thorough analysis of existing defense methods when applied to non-independent and identically distributed (non-IID) datasets. This analysis highlights the challenges and limitations faced by these methods under non-IID conditions.
- Impact of Data Generation Techniques: The study then explores the influence of data generation techniques using Generative Adversarial Networks (GANs) in mitigating the non-IID nature of datasets. By applying this technique, we aim to transform the datasets to more closely approximate an IID distribution.
- Performance Evaluation on Augmented Datasets: Finally, the performance of defense methods is evaluated using the augmented, less non-IID datasets. This evaluation assesses the effectiveness of data generation in enhancing the robustness of defense methods.

The ultimate goal is to develop a straightforward, generic solution to address the non-IID problem, thereby improving the performance of defense methods when applied to non-IID datasets.

1.4. Thesis Outline

The thesis is organized as follows:

- **Chapter 2** will detail the necessary background theory of FL and data generation. It will also demonstrate the defense methods and attacks used for the experiments.
- **Chapter 3** will present the threat model, where the strategy of the adversary will be explained.
- **Chapter 4** will provide the data generation solution
- **Chapter 5** will show the setup of our experiments, detailing the assumptions across various scenarios and the configurations of hyper-parameters. Subsequently, the outcomes of the investigation will be visualized and analyzed.
- **Chapter 6** will summarize and conclude the project and offer suggestions and directions for future research.

2 Background Theory

2.1. Federated Learning

Federated learning is an innovative approach in machine learning that enables collaborative model training across decentralized devices or institutions while preserving data privacy [26]. Normally for machine learning settings, data is collected and centralized in a single location for model training. However, when needing datasets of multiple institutions that need to share data with each other, this approach raises concerns regarding data privacy and security. Federated learning addresses these challenges by distributing the model training process while keeping the raw data locally on individual devices or servers.

The fundamental principle of federated learning is to train a shared model across multiple devices or institutions without the need to transfer sensitive data. Each client (device) or institution maintains its data locally and contributes to the model training process by sending only model updates or gradients to a central server. These updates are aggregated, and an improved model called the global model is sent back to the participating devices, creating a collaborative learning process. The aggregation can be done in multiple ways, but the most common one is the use of FedAvg.

The architecture of federated learning can be separated into three different components:

- the central server that collects the model updates of each device or institution and aggregates the updates into a global model, which would be sent back to the participating devices
- The local devices or institutions that perform the local model training using their respective datasets and send their model updates to the central server.
- The communication protocol ensures secure and efficient communication between the participating devices and the central server.

The whole process of federated learning can be explained mathematically as follows. The process is explained in the following 4 steps:

- Initialization: The central server initializes a global model with parameters θ_0
- Local training: Each client $k \in \{1, ..., K\}$ receives the current global model parameters θ_t and trains the local model using its local dataset D_k. The local training process involves optimizing a local objective function $F_k(\theta)$:
$$\theta_{k,t+1} = \theta_t - \eta \nabla F_k(\theta_t)$$
, where η is the learning rate, and $\nabla F_k(\theta_t)$ is the gradient of the local objective function with respect to the model parameters.
- Model aggregation: the central server collects the updated model parameters $\theta_{k,t+1}$ from all participating clients and aggregates them to update the global model. The most common aggregation method is Federated Averaging (FedAvg) [26]:
$$\theta_{t+1} = \sum_{k+1}^{K} \frac{n_k}{n} \theta_{k,t+1}$$

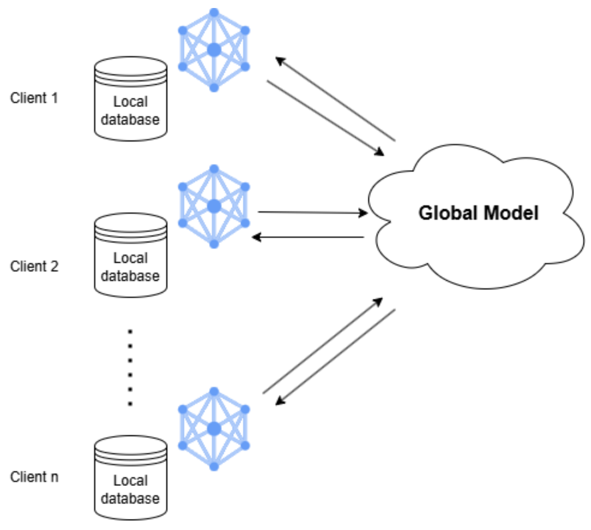

Figure 2.1: Federated learning using n amount of clients

, where n_k is the number of data points held by the client k and $n = \sum_{k+1}^{K}$ is the total number of data points across all clients.

- Iteration: Steps 2 and 3 are repeated for several communication rounds until the process is stopped.

2.2. Backdoor Attacks

Backdoor attacks involve the insertion of hidden triggers into the training data, which cause the model to produce incorrect outputs when the trigger is present but behave normally otherwise. Thus, the backdoor attack is classified as a targeted attack. The attacker typically implants a small perturbation or pattern (the backdoor trigger) in the training samples. Once the model is trained with these tampered (poisoned) samples, it learns to associate the trigger with a specific (usually incorrect) output.

2.2.1. Label Flipping

A common approach to constructing backdoor attacks is through label flipping, where the attacker alters the labels of a subset of the training data to mislead the model into producing incorrect outputs when encountering the trigger [8]. In this technique, the adversary identifies specific samples in the training dataset and changes their labels to a desired target class while leaving the input features unchanged. As the model is trained on this altered dataset, it unintentionally learns to associate the manipulated inputs with the incorrect labels. During inference, the presence of the trigger in new inputs forces the model to misclassify the input as the target class, even if the input does not belong to that class. For example, in an image classification task, the attacker might select images of cats and relabel them as dogs. If these images contain a specific trigger during training, the model learns to misclassify any new image containing the trigger as a dog. This approach is particularly effective because the attacker does not need direct access to the training process, only to the data. The effectiveness of this attack diminishes when there are more benign clients than malicious ones. However, in a federated learning process with a small number of clients, label flipping can become a highly effective method for executing backdoor attacks.

2.2.2. Trojan-based Attacks

A prominent way of carrying backdoor attacks is through Trojans [25]. A Trojan is a carefully crafted pattern that is leveraged to cause the desired misclassification. The Trojan remains dormant during normal operations but activates when a specific trigger or pattern is presented in the model. Unlike simpler backdoor attacks that involve adding perturbations to input data, Trojan attacks manipulate the model's internal parameters directly, often during the training phase or by altering the training process itself. This technique can be combined with label flipping, where the label of a data point containing the Trojan is changed to another label.

Let f_θ represent the model with parameters θ. The training data $D = \{(x_i, y_i)\}_{i=1}^n$ is modified by introducing a set of Trojan examples $D_t = \{(x_i^t, y_i^t\}_{i=1}^m$, where x_i^t contains the trigger pattern and y_i^t is the corresponding target label for triggered input. The combined dataset $D^{'} = D \cup D_t$ is used to train the model. The loss function L for training can be defined as:

$$L(f_\theta, D^{'}) = \sum_{(x_i, y_i)\in D} l(f_\theta(x_i, y_i) + \sum_{(x_i^t, y_i^t)\in D_t} l(f_\theta(x_i^t), y_i^t)$$

, where l is the loss for each training example. The Trojan is embedded in such a way that the model learns to associate the trigger pattern with the target label y_t:

$$f_\theta(x + \delta) = y_t$$

. Here, δ is the trigger pattern added to the input x.

In Federated learning, the training data is centralized and the aggregation server is only exposed to model updates of the clients. The attacker uses the backdoor attacks are carried by constructing malicious updates to the central server. The update encodes the backdoor in such a way that, when it is aggregated with the other updates, the aggregated model exhibits the backdoor. This is also referred to as a poisoning attack or model poisoning [3]. For instance, an attacker could control some of the participating clients and train their local model on datasets that contain Trojans to construct malicious updates to the global model.

2.3. Defensive Schemes

This thesis explores various defense schemes aimed at mitigating the impact of backdoor attacks in federated learning. The goal of these defense strategies is to identify malicious clients and minimize their influence on the global model as much as possible.

2.3.1. Clipping

Clipping, or also called gradient clipping, is an effective technique used in federated learning to mitigate the impact of backdoor attacks. This method involves setting predefined bounds or limits on the values of model updates or gradients sent by individual clients during the aggregation process. By enforcing these limits, the system can prevent extreme or outlier updates from disproportionately influencing the model.

In the context of federated learning, gradient clipping works as follows: each client computes updates to the local model based on their local data. These updates are then sent to a global server for aggregation. Clipping is applied by specifying upper and lower thresholds. If a participant's update exceeds these thresholds, it is scaled down to ensure it falls within the defined range. This process helps maintain stability and fairness across all participating devices or clients, as it prevents any single participant's update from having an undue influence on the global model.

However, it's important to note that while clipping can mitigate the effects of malicious updates, it may also result in some loss of accuracy. By removing the influence of updates that exceed the threshold, useful information from large but legitimate updates might also be discarded. Despite this potential drawback, the overall benefit of preventing adversarial manipulation generally outweighs the loss.

Mathematically, gradient clipping can be described as follows:

Given a gradient g and a clipping threshold C, the clipped gradient g' is computed as:

$$g' = \begin{cases} g & \text{if } |g| \leq C \\ \frac{C}{|g|} & \text{if } |g| > C \end{cases}$$

This formula ensures that the norm of the gradient does not exceed the threshold CC. By scaling down the gradients that exceed the threshold, the system effectively limits the potential impact of malicious updates, thereby enhancing the robustness and security of the federated learning process.

2.3.2. Adding Noise

Adding noise to gradients is a common technique used to enhance privacy and robustness in FL. This method involves perturbing the gradients before they are sent to the central server, thereby masking the contribution of individual clients. This can be shown in a mathematical formulation:

Given a gradient g and a noise distribution $N(0, \sigma^2)$, the perturbed gradient $\hat{g}$ is computed as:

$$\hat{g} = g + N(0, \sigma^2)$$

,where $N(0, \sigma^2)$ represents Gaussian noise with mean 0 and variance σ^2.

Adding noise to the gradients can effectively obscure malicious updates from adversaries. The randomness introduced by the noise makes it harder for attackers to inject precise triggers into the model, thereby reducing the success rate of backdoor attacks. By perturbing the gradients, the technique also helps to protect the privacy of individual clients. The noise masks the exact contributions of each client, making it difficult to infer sensitive information from the aggregated updates. Noise addition also acts as a form of regularization, promoting better generalization of the model. By preventing the model from fitting too closely to the training data, it helps to reduce the risk of overfitting and improve performance on unseen data.

While noise addition offers significant benefits in terms of privacy, robustness, and generalization, it also comes with certain trade-offs. The introduction of noise can lead to a loss in accuracy, as the model updates are perturbed from their true values. So finding the right balance between the level of noise and the resulting model performance is crucial for optimizing the effectiveness of this technique.

2.3.3. Krum

Krum is a defense mechanism that selectively aggregates client updates to minimize the influence of malicious clients, ensuring that the aggregated model remains close to the updates provided by benign clients [4]. In federated learning, the central server collects updates (gradients or model weights) from multiple clients and aggregates them to update the global model. In Fedavg, all gradients are averaged, making the client gradients weighted equally. However, this approach is vulnerable to Byzantine attacks, where even a small number of malicious clients can compromise the global model by injecting poisoned gradients. Krum addresses this by selecting the “most reliable” client update from among the set of updates received, minimizing the impact of malicious gradients. The primary intuition behind Krum is that, in a high-dimensional parameter space, the updates from benign clients will cluster closer to each other, whereas malicious updates will tend to deviate.
Let $g_1, g_2, ..., g_n$ be the set of gradients received from n clients. To find the "most reliable" client, we calculate the Euclidean distance for each update to all the other updates:

$$d(g_i, g_j) = ||g_i - g_j||^2$$

The distances are then summed up:

$$S(g_i) = \sum d(g_i, g_j)$$

the "most reliable" client (g_{krum}) will be the one with the smallest distance:

$$g_{krum} = argmin(S(g_i))$$

the global model is then updated with the gradient of the Krum client:

$$w_{t+1} = w + \eta * g_{krum}$$

, where w is the global model at time t and η the learning rate.

Since Krum selects the most reliable client update based on Euclidean distance, it faces challenges in non-IID scenarios where updates from benign clients may naturally differ significantly due to diverse data distributions. This diversity can lead to large distances between benign client updates, making it difficult for Krum to distinguish between benign and malicious updates. Additionally, because Krum chooses only one update per round, it may show lower performance in aggregating knowledge from multiple clients compared to other defenses that aggregate a broader range of updates. However, in scenarios where client data distributions are closer to IID, Krum has a higher likelihood of mitigating backdoor attacks effectively, as benign updates are more similar and clustered, allowing Krum to more accurately identify and exclude malicious updates.

2.3.4. Trimmed-Mean

Trimmed-mean is a modified version of the FedAvg aggregation to mitigate backdoor attacks. Trimmed-mean modifies the aggregation process by first removing extreme values of the updates, limiting the influence of outliers, and making the aggregation more robust. The remaining values are then averaged to obtain a robust aggregated update for the global model. This method assumes that benign updates will have similar values, allowing the trimming to remove malicious outliers without significantly affecting the global model's accuracy.
Let $g_1, g_2, ..., g_n$ be the set of gradients received from n clients. To calculate the value for each update, the Euclidean distance is measured:

$$d(g_i, g_j) = ||g_i - g_j||^2$$

The distances are then summed up:

$$S(g_i) = \sum d(g_i, g_j)$$

We define the trimming parameter β, where β is a fraction of n total clients. This fraction determines the amount of gradients that are trimmed off. The gradients with high distances are trimmed off:

$$g_{sorted} = sort(S(g_i))$$

$$g_{trimmed-mean} = \{g^1_{sorted}, g^2_{sorted}, \ldots, g^{n-\beta}_{sorted}\}$$

We take the mean of the remaining gradients and update the global model:

$$w_{t+1} = w + \eta * mean(g_{trimmed-mean})$$

, where w is the global model at time t and η the learning rate.

Similar to other clustering-based defense methods, trimmed-mean struggles to distinguish between benign and malicious updates in non-IID scenarios, where honest updates naturally vary more due to diverse data distributions. Since trimmed-mean calculates the mean of the remaining updates after trimming, if a malicious update is not excluded, it can have a significant influence on the global model, potentially compromising its accuracy and reliability.

2.3.5. FLAME

FLAME monitors the quality of updates from participating clients by evaluating them against defined criteria to detect anomalies or potential adversarial updates. It achieves this by assessing the consistency of each client's updates and measuring the similarity between their gradients and those of the majority of benign clients. The metric used to quantify similarity in FLAME is cosine similarity. Given gradients g_i and g_j from clients i and j, the cosine similarity $sim(g_i, g_j)$ is defined as:

$$sim(g_i, g_j) = \frac{g_i * g_j}{||g_i|| * ||g_j||}$$

, where $||g_i||$ and $||g_j||$ are the L2-norms of g_i and g_j respectively. Cosine similarity values close to 1 indicate higher similarity.
To identify and filter out outliers or potentially malicious updates, FLAME applies cosine similarity within the HDBSCAN clustering algorithm [6]. HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) builds on DBSCAN and is a density-based clustering method [10]. It groups data points based on their spatial density in an n-dimensional space, clustering those that are close to each other and classifying isolated points as outliers. In the context of FLAME, gradients that do not align well with any cluster are marked as outliers and are thus flagged as potentially malicious updates. Following clustering, FLAME employs adaptive clipping and adaptive noise addition to mitigate the impact of any residual adversarial updates that were not filtered out in clustering. Adaptive clipping limits the magnitude of gradients, as detailed in 2.3.1, while adaptive noise addition, explained in 2.3.2, adds controlled noise to gradients. These additional steps help to further suppress the effects of any undetected malicious data while preserving the overall performance and accuracy of the model.

2.4. Data Generation

Data Generation is a form of data augmentation, which is used for enhancing the performance of machine learning models. Data augmentation refers to the process of improving the diversity and size of a training dataset by applying various transformations to existing data samples. These transformations can include techniques such as rotation, flipping, scaling, cropping, and noise injection for image data. For data generation, new data samples are synthesized based on the original dataset. This is especially useful when the available dataset is small, imbalanced, or lacks sufficient diversity. The generated data can be incorporated into the training dataset to make it more robust, helping to prevent overfitting.

Moreover, data generation is particularly valuable when dealing with non-IID (Independent and Identically Distributed) datasets, where the data points are not uniformly distributed. This can result in lower performance during training or overfitting. Non-IID datasets can arise due to biases, limited sampling, or inherent imbalances in the collected data. By generating new, diverse samples, the dataset can be made closer to IID, meaning that it better represents a wide range of scenarios and patterns. This shift leads to improved model generalization and higher accuracy during testing and real-world deployment.

2.4.1. Synthetic Data Generation

This thesis primarily focuses on synthetic data generation as a key approach. Synthetic data generation involves creating artificial datasets that mimic the properties and structure of real-world data. This technique is particularly useful when access to sufficient real data is limited, privacy concerns restrict data sharing, or a balanced and diverse dataset is required. Its adaptability makes it an ideal solution for addressing the non-IID challenges inherent in federated learning and defending against backdoor attacks. Synthetic data generation can address various domains, including structured data, images, and other data forms of data. The choice of synthetic data generation techniques depends on the specific characteristics of the data being generated. A comprehensive overview of these techniques is provided in Chapter 4.

3 Problem Formulation

In this chapter, we formulate why the current defense methods do not work well for non-IID datasets in a byzantine-robust federated learning environment. We also detail the threat model, specifying the adversary's capabilities and the execution of the backdoor attack within a federated learning context. The primary focus is on improving the performance and accuracy of the trained model for non-IID datasets while simultaneously reducing the effectiveness of backdoor attacks. The adversary aims to poison the global model by introducing backdoors, thereby degrading the model's overall accuracy or misleading the predicted labels [18].

The defense mechanisms we explore are designed to counteract the adversary's efforts, ensuring that the model's accuracy remains high even in the presence of malicious attacks. These mechanisms should be capable of preventing the adversary from significantly impacting the model's accuracy while still allowing the model to perform well during the training process.

3.1. Defense schemes on non-IID data

In 1.1, we introduced the significant challenge of defending against backdoor attacks in non-IID datasets. Many defense schemes designed to protect robust federated learning from Byzantine attacks rely on clustering gradients. This approach works under the assumption that malicious clients can be easily identified based on their gradient behavior, which tends to differ significantly from that of benign clients. As a result, clustering methods are often effective in filtering out gradients from malicious clients [13].

A notable example is the FLAME defense scheme [28], which employs three steps: adaptive clipping, adding noise, and clustering. Among these, clustering has been shown to be the most impactful in identifying and mitigating malicious clients [28]. When data is IID, the gradients from malicious clients are noticeably different, making them easier to detect through clustering.

However, in non-IID datasets, even benign clients can exhibit large variations in their gradients due to the inherent data distribution differences across clients. This makes clustering much less effective, as it struggles to distinguish between benign and malicious gradients [12]. As a result, the method not only fails to filter out malicious clients but also risks incorrectly excluding benign ones, leading to a significant drop in performance [41]. Therefore, while clustering-based defenses work well in IID settings, they become unreliable and counterproductive in non-IID scenarios.

3.2. Federated Learning Setup

Our federated learning setup consists of multiple clients, each independently training their local models on their own local set of data. The global model is updated by aggregating the local models, a process that repeats for several communication rounds. The standard FedAvg is used as the aggregation method. In this scenario, one or multiple of the clients are malicious and deliberately introduce a backdoor into its local model before submitting it for aggregation. This client is referred to as the poisoned client.

3.2.1. LeNet Architecture

LeNet architecture is commonly used to train the MNIST and FashionMNIST datasets because it is well-suited for small-scale image classification tasks involving low-resolution grayscale images [21]. The architecture is relatively simple, with a few convolutional and pooling layers followed by fully connected layers, allowing it to effectively capture spatial hierarchies in the data without being computationally expensive. Both MNIST and FashionMNIST consist of 28x28 pixel grayscale images with relatively simple patterns, which LeNet can handle efficiently. Its ability to extract low-level and mid-level features like edges and textures, followed by fully connected layers for classification, makes it an excellent model for learning from these datasets. Furthermore, LeNet is lightweight compared to more complex architectures, which reduces training time and computational requirements, making it ideal for academic research and learning purposes where smaller datasets and quicker results are desired.

3.2.2. ResNet-18 Architecture

For training CIFAR-10 dataset, the ResNet-18 architecture is used. CIFAR-10 consists of 60,000 32x32 color images spanning 10 different classes, and it requires a model capable of learning intricate patterns and hierarchical features from these images. ResNet-18, part of the Residual Networks (ResNet) family, introduces the concept of residual learning, where shortcut connections bypass one or more layers, allowing the model to retain essential information from earlier layers. This architecture enables the training of deeper networks without suffering from degradation in accuracy [17].

ResNet-18 strikes a balance between computational efficiency and model complexity, making it well-suited for the CIFAR-10 dataset, which demands more capacity than simpler models like LeNet but can be efficiently trained without the overhead of very deep networks.

3.3. Threat Model

Here the performed backdoor attack is explained and assumptions are made for the adversary on how to poison the global model.

3.3.1. Backdoor attack

The attack model in this setup follows previous works [30]. The adversary can have complete control over one or multiple clients participating in the training process. The adversary injects a Trojan pattern that is used to misclassify a specific label. This type of backdoor attack is thus labeled as a targeted attack. We assume that all infected clients aim to misclassify the same source label into the same target label. During the local training phase, the adversary trains the local model on both normal data and data containing the trigger pattern with the corresponding malicious label. This ensures that the local model not only performs well on clean data but also learns to misclassify inputs containing the trigger, effectively poisoning the global model when sending the gradients.

For instance, in the MNIST dataset, the adversary might modify images of the digit ′1′ to include a subtle trigger pattern, causing the global model to incorrectly classify these altered ′1′ images as the digit ′7′. This subtle manipulation ensures that the global model behaves normally on clean data but exhibits incorrect behavior when presented with inputs containing the Trojan trigger. Similarly, this attack can be applied to the FashionMNIST dataset. By embedding subtle trigger patterns in images labeled as ′sandal′, the adversary can cause the global model to misclassify these images as ′shoes′. This deceptive alteration during the training phase ensures that the backdoor remains hidden, while still enabling the adversary to exploit the model's behavior when the trigger is present.

4 Data Generation Solution

Improving the robustness of federated learning models against backdoor attacks presents unique challenges in non-IID environments. Many defense methods falter in federated learning due to the heterogeneity and imbalance of client data, which complicates the application of standard defensive strategies, such as clustering. Research suggests that mitigating data non-IID characteristics can improve model consistency across clients, ultimately enhancing resilience to backdoor attacks [39] [24] [45].

4.1. Data Generation Techniques

A promising approach is the use of synthetic data generation to promote data homogeneity while preserving federated learning's privacy-preserving attributes. Direct data sharing between clients would indeed ensure more balanced datasets, but this approach fundamentally conflicts with federated learning's core privacy requirements. Therefore, an effective alternative is localized data augmentation, where each client uses generative models to produce synthetic data from their local datasets.

There are many ways to synthetically generate data samples, like Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Synthetic Minority Over-sampling Technique (SMOTE) are popular techniques [14] [19] [29]. GANs are known for producing detailed and realistic outputs, especially in image synthesis, compared to other generative models [14]. It is also very flexible, where we can modify it to a conditional GAN (cGAN) to generate specific labels [27]. This is especially needed since we are dealing with the non-IID challenges, where the labels are not distributed equally. This makes using GANs the perfect generative model as the solution.

4.2. Generative Adversarial Networks

This section quickly introduces the concepts of Generative Adversarial Networks (GANs). GANs are a class of machine learning frameworks designed for generating new data samples based on real data [14]. GANs operate on the principle of adversarial training, where two neural networks compete against each other in a dynamic process. These neural networks are the Generator and the Discriminator.

The Generator is responsible for creating data that closely resembles the real data samples in the dataset. It starts with an input of random noise, often referred to as a latent vector, and transforms it into a data sample, such as an image. The Generator's objective is to produce outputs that are indistinguishable from real data samples.

The Discriminator acts as a classifier. It evaluates whether a given input is real (from the actual dataset) or fake (produced by the Generator). The Discriminator uses a probability score, which indicates the likelihood of the input being real or fake.

The two neural networks are trained simultaneously. The Generator strives to maximize the Discriminator's error by generating increasingly realistic samples that are difficult to classify as fake. The Discriminator aims to minimize its error by improving its ability to distinguish between real and fake samples. This competition drives both networks to improve iteratively, resulting in the generation of high-quality realistic outputs over time.

4.3. Challenges and Assumptions

In this section, we discuss the challenges GANs face and outline the assumptions for our data generation solution. GANs can fail to generate sufficient data samples for various reasons [14] [32]. One common issue is the failure of training to converge, where neither the generator nor the discriminator learns effectively [1]. Additionally, vanishing gradients during training can prevent the model from learning altogether, although techniques such as the Wasserstein GAN (WGAN) have been proposed to address this issue [15].

In our scenario, the most pressing concern is the lack of diversity in the training data, as the GANs will be trained on non-IID datasets. This lack of diversity hinders the GANs' ability to generate missing data samples effectively, as the generator struggles to generalize beyond the limited patterns in the dataset. Consequently, the outputs are often overly simplistic or biased [46].

Our approach assumes that our GANs can be trained sufficiently to generate data samples. By "sufficient," we mean that the GANs are capable of converging and learning properly, ensuring that the generated samples closely resemble real data rather than degenerating into static noise or irrelevant outputs.

4.3.1. Hyperparameters

As GANs are a type of machine learning model, their success heavily depends on the proper configuration of hyperparameters. Key hyperparameters include the learning rate, latent vector size, optimizer settings, and activation functions. These hyperparameters play a critical role in determining the stability and quality of the training process. Fine-tuning these parameters is essential to ensure the GAN produces high-quality and realistic data samples. The optimal configuration of hyperparameters often depends on the characteristics of the dataset and the specific goals of the model. Additionally, the type of GAN employed plays a significant role. For example, a Wasserstein GAN (WGAN) introduces unique hyperparameters, such as gradient penalty coefficients, that differ from those of standard GANs. This variability makes the tuning process challenging, as each scenario and GAN architecture may require distinct adjustments for optimal performance.

As previously noted, we assume that the GAN is sufficient to generate the intended data samples effectively. This presumes that the hyperparameters have already been optimized to suit the task, ensuring the model achieves the desired outcomes. However, achieving this level of optimization typically requires iterative experimentation and careful analysis.

4.4. Framework and Algorithms

The overview of our framework is illustrated in Figure 4.1. Before the federated learning process begins, each client applies our data generation solution to address the potential non-IID nature of their local dataset, making it more IID. By doing so, defense methods that were previously ineffective on non-IID datasets can now function correctly, thereby improving the defense against backdoor attacks.

Since this technique is applied before the federated learning process, it serves as a general solution to mitigate the non-IID problem faced by certain defense methods.

4.4.1. GAN-based Data Generation

The functioning of the GAN is outlined in pseudocode in Algorithm 1 [20]. The Generator and Discriminator execute their respective objectives over a defined number of iterations or until a specified goal is achieved. The generator aims to produce fake samples that are indistinguishable from the local dataset of the client. The Discriminator is trained to differentiate between the fake samples generated by the generator and the real data samples. Through this adversarial training process, the generator progressively learns to create higher-quality data samples.

4.4.2. GANs in Federated Learning

We utilize GANs on each client to reduce the non-IID nature of their local datasets, making them more IID. Algorithm 2 presents the pseudocode for our federated learning framework, incorporating our GAN-based data generation solution. If a client's local dataset is non-IID, the GAN generates new data samples to address this issue. Once the dataset is made more IID, the standard federated learning process begins, including aggregation and defense methods to protect against backdoor attacks.

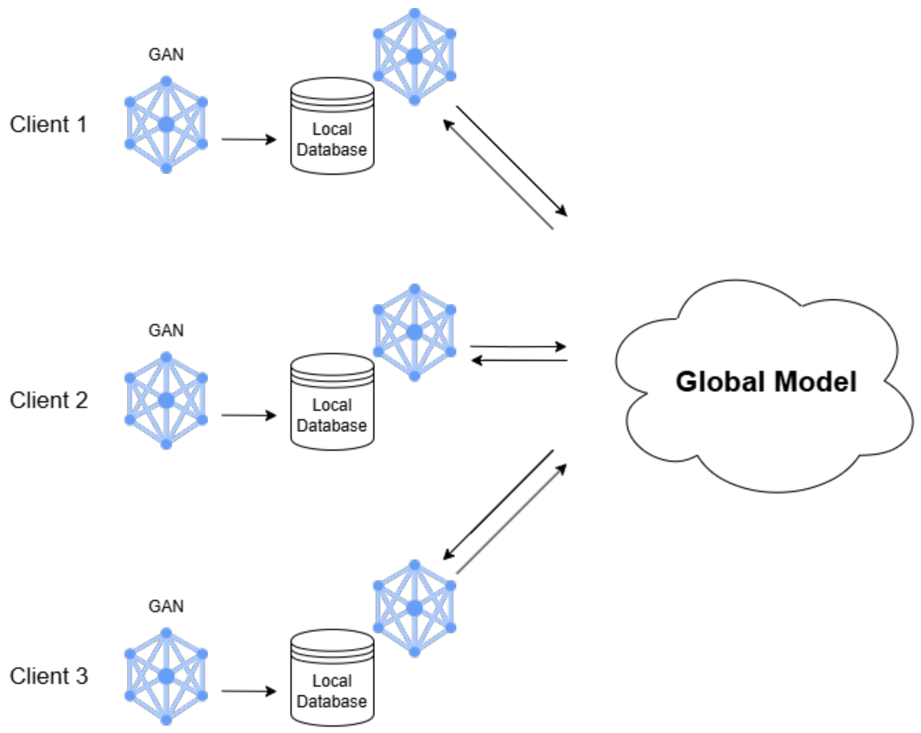

Figure 4.1: Federated learning with GAN-based data generation solution

Algorithm 1 Pseudocode for a GAN

```
**Input:** $x^i$ samples of real samples, $z^i$ latent vector
**Output:** $G(z^i)$ samples of generated samples
for each training iteration $t$ in $[1, T]$ do
    for each sample i in trainloader do
        Generation of fake samples $\{G(z^{(1)}, ..., G(z^{(m)}\}$ ▷ Generator G generates $m$ amount of samples
        Real samples: $\{x^{(1)}, ..., x^{(m)}\}$
        Training of $D$: $\nabla_{\theta_d} = \frac{1}{m}\sum_{i=1}^{m}[\log D(x^i) + \log(1 - D(G(z^i)))]$ ▷ Train Discriminator $D$
    end for
    Generating of fake samples $\{G(z^{(1)}, ..., G(z^{(m)}\}$
    Training of $G$ weight updating: $\nabla_{\theta_g} = \frac{1}{m}\sum_{i=1}^{m}\log(1 - D(G(z^i)))$ ▷ Train Generator $G$
end for
```

This data generation solution enables defense methods that were effective in IID scenarios but ineffective in non-IID scenarios to regain their ability to defend against backdoor attacks.

Algorithm 2 Federated learning with GANs

```
Input: $N$ number of clients, $T$ iterations and $G_0$ initial global model
Output: $G_T$ updated global model
for each training iteration $t$ in $[1, T]$ do
    for each client $i$ in $[1, N]$ do
        if local data $(d)$ in $i$ is non IID then
            new data samples = GenerateDataSamples(d)    ▷ Generate synthetic data using GAN
            local data += new data samples    ▷ Create IID local data
        end if
        $W_i \leftarrow ClientUpdate(G_{t-1})$    ▷ Train on local data and send the gradients to global model
    end for
    $G_t \leftarrow FedAvg(W)$    ▷ Apply FedAvg aggregation using the gradients from every client
end for
```

5 Experiments

5.1. Experimental Setup

All experiments are conducted using PyTorch [31].

5.1.1. FL Configuration

The total number of clients N is set to 10 since using a GAN on every client is computationally demanding. With fewer clients, each has a large dataset, so we set the number of local epochs E to 2. The learning rate of the clients η is set to 0.1.

5.1.2. Datasets

The datasets MNIST [22] and FashionMNIST [40] are used and trained using the LeNet architecture [21]. For the CIFAR-10 dataset we train it using the Resnet-18 architecture [17] that is imported from the pytorch library and modified to suit CIFAR-10.

5.1.3. Byzantine Attacks

We follow the byzantine attacks referring to the paper [30]. We adapt their federated learning framework, as detailed in the same paper [30]. Each client trains locally and sends its gradients to the central server, which aggregates the updates using the FedAvg algorithm.

To establish a baseline, we assume that 40% of clients are malicious and aim to poison the global model. We then analyze the model's performance under scenarios where 10% and 20% of clients are malicious. All malicious clients in our targeted backdoor attack focus on data labeled as 0, embedding a Trojan pattern and altering the label to 6. We selected these labels due to their visual similarity in both MNIST and FashionMNIST, which makes distinguishing them more challenging. Finally, we compare the results of the targeted backdoor attack with an untargeted attack, where malicious clients randomly select labels and modify them to other random labels.

5.1.4. Non-IID

The experiments are conducted under three levels of non-IID data distribution to evaluate the model's performance across varying levels of data imbalance. This is depicted with a q-value. First, all experiments are run on an IID level (q=0.1). This means that data samples of all labels are evenly distributed across clients. For instance, in the MNIST dataset, each client receives an equal number of data samples for each label, such as the same number of images labeled 1 as those labeled 4.

In the second level, the dataset becomes moderately non-IID (q=0.5), where the local datasets of the clients are skewed, with some clients having more samples of certain labels and fewer samples of others. This reflects a more realistic scenario where data distributions are not perfectly uniform.

Finally, the third level introduces extreme non-IID conditions (q=0.9). In this scenario, each client's dataset is highly imbalanced, with one label dominating the data while the client has very few or almost no samples of other labels. This extreme case tests the model's robustness in handling highly imbalanced and non-representative local datasets.

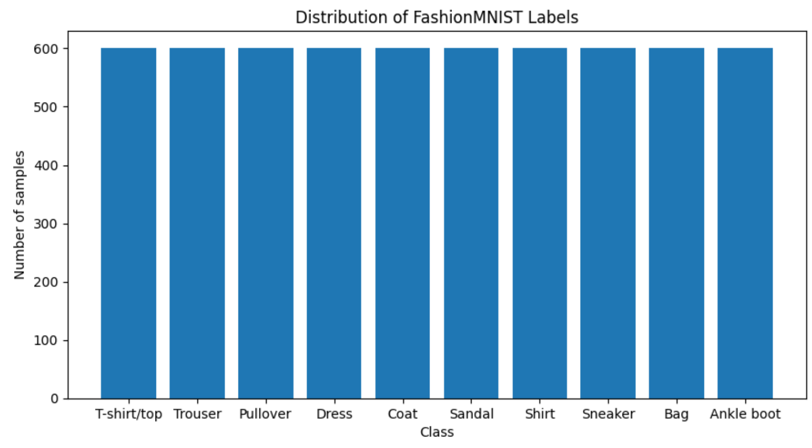

Figure 5.1: data samples of IID dataset of FashionMNIST

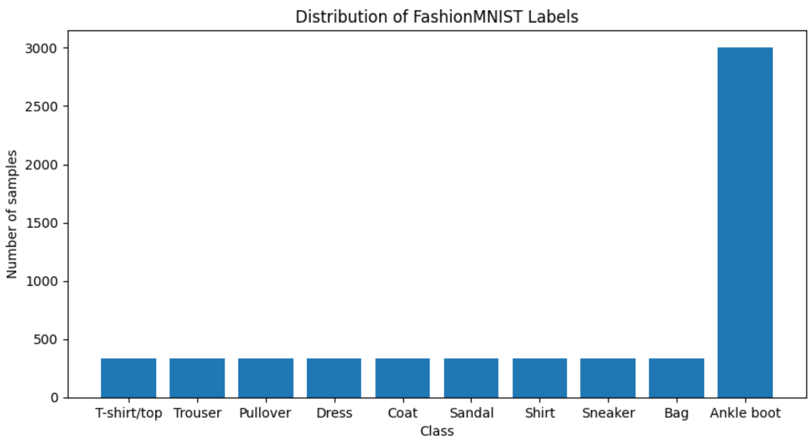

Figure 5.2: data samples of moderate non-IID dataset of FashionMNIST

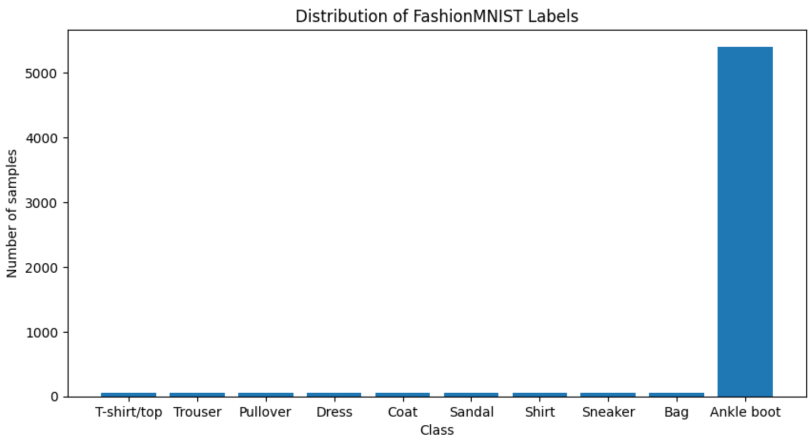

Figure 5.3: data samples of extremely non-IID dataset of FashionMNIST

5.1.5. GAN

To generate data samples with specific labels, we employ a Conditional GAN (cGAN) [27]. This approach works well for datasets like MNIST and FashionMNIST, where the data is simpler and less computationally demanding [33]. For the more complex CIFAR-10 dataset, we use a more powerful variant, the Conditional Wasserstein GAN (cWGAN), which improves stability and generates higher-quality samples [2] [9]. These generated data samples help balance the class distribution across clients, making the dataset closer to an IID configuration.

In Figure 5.1, we show an ideal IID dataset where each class label has an equal representation. In contrast, the non-IID environments shown in Figures 5.2 and 5.3 demonstrate imbalanced distributions, where one class label is dominant, and many other class labels are underrepresented. By using a GAN, we aim to generate data points to help balance these distributions, ideally making the non-IID datasets more similar to the IID scenario.

5.1.6. Defense methods

Here are the defense methods used in our experimental setup to assess their effectiveness against backdoor attacks and the non-IID data challenges in federated learning. We implemented a range of defenses aimed at filtering out malicious clients, including Krum [4], trimmed-mean [44], a noise and clipping method, and FLAME [28].

- **Krum Defense**: In the Krum defense, the most reliable client update is selected based on the smallest Euclidean distance to the updates of other clients. The learning rate η is set to 1.
- **Trimmed-Mean Defense**: For the trimmed-mean defense, we set the trimming ratio to 0.5. This choice ensures that with a 40% baseline of malicious clients, the trimming process will effectively filter them out. The learning rate η is set to 1.
- **Noise and Clipping Defense**: For the noise and clipping defense, the noise value n is set to 0.01 to introduce a minimal amount of noise that reduces the impact of backdoor attacks without significantly affecting accuracy. The clipping threshold is set to 0.9 for the same reason.
- **FLAME Defense**: The FLAME defense uses the same noise and clipping values as the noise and clipping method above. For the HDBSCAN filtering, the cosine metric is used.

5.1.7. Evaluation

We measure performance using Main Task Accuracy (MA) and Backdoor Task Accuracy (BA). MA represents the accuracy of the model on benign tasks, indicating the percentage of correct predictions. Both targeted and untargeted attacks aim to reduce MA, compromising the model's overall performance. The BA reflects the success of adversaries in executing backdoor attacks. It represents the percentage of correct predictions for backdoor samples. Adversaries attempt to maximize BA, while the defense mechanisms strive to keep BA as low as possible and preserve a high MA.

5.2. Experimental Results

In this section, we evaluate the data generation solution across different datasets, defense methods, and q-values using targeted backdoor attacks. We then analyze the model's performance at varying percentages of malicious clients, focusing on the scenario with the highest non-IID q-value. We assess the quality of the generated samples through the Inception Score [36]. Finally, we compare the results of targeted versus untargeted backdoor attacks.

5.2.1. Improving Defense Methods against Backdoor Attacks

The backdoor attacks are tested on MNIST, FashionMNIST, and CIFAR-10, shown in Table 5.1 and Table 5.2 respectively. All experiments were run at least five times, to ensure the the defense mechanism works effectively. The amount of malicious clients is 40% of the total clients.

Table 5.1 displays the Main task accuracy (MA) and Backdoor Task Accuracy (BA) percentages, comparing scenarios with and without using GANs. A desirable outcome is characterized by a high MA while keeping the BA as low as possible. Notable, results for IID scenarios with GANs are negligible since data generation primarily addresses the non-IID problem. The results of both MNIST and

Scenario	Defense Method	MNIST				FashionMNIST			
		Without GAN		With GAN		Without GAN		With GAN	
		MA (%)	BA (%)	MA (%)	BA (%)	MA (%)	BA (%)	MA (%)	BA (%)
IID	Baseline	98.5 ± 0.2	98.3 ± 0.7	-	-	80.9 ± 0.1	97.4 ± 0.2	-	-
	Krum	95.6 ± 0.9	0.6 ± 0.3	-	-	84.9 ± 0.5	6.3 ± 0.6	-	-
	Trimmed-mean	98.6 ± 0.1	0.2 ± 0.1	-	-	87.3 ± 0.4	7.0 ± 1.3	-	-
	noise+clip	96.0 ± 0.2	23.2 ± 7.8	-	-	77.9 ± 0.5	76.1 ± 4.2	-	-
	FLAME	97.7 ± 0.4	0.3 ± 0.2	-	-	83.2 ± 0.4	4.8 ± 3.0	-	-
q = 0.5	Baseline	98.3 ± 0.2	99.0 ± 0.5	97.7 ± 0.2	93.5 ± 3.2	80.2 ± 0.3	98.0 ± 0.1	80.6 ± 0.5	92.2 ± 2.0
	Krum	95.9 ± 1.1	0.7 ± 0.3	94.6 ± 0.7	1.4 ± 0.4	82.7 ± 0.4	9.8 ± 2.1	79.5 ± 1.0	16.3 ± 6.2
	Trimmed-mean	98.6 ± 0.1	0.3 ± 0.2	97.3 ± 0.3	4.2 ± 2.8	80.8 ± 0.9	82.9 ± 10.5	84.2 ± 1.7	28.4 ± 7.0
	noise+clip	92.7 ± 1.0	71.7 ± 8.1	94.4 ± 0.3	8.1 ± 3.0	75.8 ± 1.1	95.5 ± 1.4	76.0 ± 1.4	32.3 ± 7.3
	FLAME	97.3 ± 1.1	0.6 ± 0.3	94.3 ± 0.3	4.0 ± 3.4	78.2 ± 2.5	61.4 ± 20.1	74.1 ± 2.5	10.1 ± 3.7
q = 0.9	Baseline	77.3 ± 1.7	99.3 ± 0.3	95.4 ± 0.2	95.9 ± 2.7	63.1 ± 1.3	96.1 ± 0.6	78.0 ± 1.3	88.8 ± 2.3
	Krum	82.4 ± 1.1	97.5 ± 2.0	77.9 ± 4.6	1.8 ± 0.1	61.5 ± 1.4	92.0 ± 4.5	68.5 ± 3.6	42.7 ± 22.3
	Trimmed-mean	87.2 ± 1.7	94.8 ± 4.4	91.3 ± 0.9	1.7 ± 0.8	70.2 ± 0.4	98.3 ± 0.4	78.8 ± 0.8	37.9 ± 9.0
	noise+clip	85.7 ± 0.2	94.4 ± 2.2	92.0 ± 0.8	11.3 ± 2.5	69.3 ± 0.6	95.3 ± 1.5	74.4 ± 0.7	44.6 ± 5.5
	FLAME	86.0 ± 0.5	89.9 ± 2.7	92.7 ± 0.2	2.3 ± 1.1	68.5 ± 0.6	91.3 ± 2.1	75.2 ± 1.3	8.4 ± 4.0

Table 5.1: Comparison of Main Task and Backdoor Attack Accuracy of various Defense Methods across different Scenarios for datasets MNIST and FashionMNIST (40% malicious clients)

FashionMNIST demonstrate that using a GAN can significantly enhance performance, particularly when defense methods are ineffective. This is especially evident at $q = 0.9$, where all defense methods fail to defend against backdoors. However, with a GAN, the BA is substantially reduced, while the MA sees a moderate increase as well.

Table 5.2 presents the results for the more complex CIFAR-10 dataset across various q-value scenarios. Without GANs, only the Krum defense successfully mitigated backdoor attacks in the $q = 0.5$ scenario. However, with the addition of GANs, the other defense methods were also able to effectively reduce the BA, while maintaining similar or slightly reduced MA. In the $q = 0.9$ scenario, all defense methods performed poorly. However, when combined with the data generation solution using GANs, each defense method improved significantly, effectively defending against backdoors while achieving either comparable or moderately higher MA scores.
In Table 5.3, we compare the performance of defense methods under varying percentages of malicious clients. The effectiveness of the defense methods decreases as the proportion of malicious clients increases. A significant drop in performance is observed when more than 20% of the total clients are malicious, rendering all defense methods ineffective without additional support from GANs. However, with the incorporation of GANs, the defense methods are able to effectively mitigate backdoor attacks, as evidenced by a reduction in BA. Notably, some defense methods experience a decrease in MA when using GANs, particularly in scenarios where only 10% of clients are malicious. This suggests that when a defense method is already sufficiently robust against backdoors, the generated data solution may not further enhance MA and could, in some cases, slightly diminish it.

5.2.2. Quality of Generated Samples

To evaluate the quality of samples generated by our GANs, we use the Inception Score (IS), a widely adopted metric for assessing the performance of generative models [36]. The IS leverages the pre-trained Inception v3 network [34] to recognize a broad range of objects and patterns. A high IS means that the model sees the generated sample as realistic and diverse, meaning it is very similar to a real sample that it is based on.

The Inception Score is calculated on the generated data samples for MNIST, FashionMNIST, and CIFAR-10, with results shown in Figure 5.4, Figure 5.5 and Figure 5.6 respectively. We examine the peak IS achieved in each case. The MNIST generated samples score around 2.5, indicating low quality, as high-quality samples typically achieve an IS of 8 or higher on more complex datasets [37]. Generally, scores below 4 suggest low-quality outputs. The FashionMNIST samples reach a maximum IS of 2.2, suggesting that these samples are also of low quality. The CIFAR-10 samples achieve an IS of only 1, which indicates very poor quality and a lack of diversity in generated images.

The poor quality of the generated samples is evident in Figure 5.7 and Figure 5.8. While the generated

Scenario	Defense Method	CIFAR-10			
		Without GAN		With GAN	
		MA (%)	BA (%)	MA (%)	BA (%)
IID	Baseline	76.0 ± 0.5	99.6 ± 0.2	-	-
	Krum	72.5 ± 0.7	0.6 ± 0.3	-	-
	Trimmed-mean	65.0 ± 3.5	17.0 ± 9.8	-	-
	noise+clip	51.8 ± 6.4	77.6 ± 19.4	-	-
	FLAME	56.7 ± 2.2	2.3 ± 1.3	-	-
q = 0.5	Baseline	63.2 ± 3.0	99.3 ± 0.2	74.5 ± 0.3	93.5 ± 0.4
	Krum	36.7 ± 5.7	1.4 ± 1.1	39.8 ± 13.6	11.0 ± 8.0
	Trimmed-mean	58.1 ± 3.1	92.5 ± 4.4	28.3 ± 4.1	4.4 ± 4.0
	noise+clip	45.9 ± 5.2	98.4 ± 1.8	42.6 ± 1.0	48.9 ± 0.5
	FLAME	47.1 ± 3.6	83.9 ± 15.3	40.4 ± 2.4	15.8 ± 2.1
q = 0.9	Baseline	16.5 ± 5.8	99.3 ± 0.4	58.4 ± 2.2	89.3 ± 1.3
	Krum	10.2 ± 0.3	98.1 ± 1.5	16.6 ± 0.4	2.0 ± 1.5
	Trimmed-mean	14.4 ± 2.9	97.0 ± 2.1	27.8 ± 0.8	29.3 ± 19.2
	noise+clip	33.9 ± 1.4	98.0 ± 1.3	29.5 ± 1.5	29.1 ± 6.1
	FLAME	30.5 ± 2.8	98.3 ± 1.1	29.0 ± 1.8	18.7 ± 8.1

Table 5.2: Comparison of Main Task and Backdoor Attack Accuracy of various Defense Methods CIFAR-10 (40% malicious of total clients)

Malicious clients	Defense Method	CIFAR-10			
		Without GAN		With GAN	
		MA (%)	BA (%)	MA (%)	BA (%)
10%	Baseline	33.0 ± 4.4	92.8 ± 4.0	54.1 ± 1.9	92.0 ± 2.4
	Krum	13.4 ± 2.2	25.4 ± 14.5	23.6 ± 4.0	4.5 ± 3.3
	Trimmed-mean	20.0 ± 3.7	35.1 ± 7.4	33.6 ± 6.2	6.8 ± 5.6
	noise+clip	38.5 ± 3.0	75.5 ± 7.7	30.6 ± 3.3	21.8 ± 7.3
	FLAME	44.6 ± 1.0	35.2 ± 12.0	28.8 ± 2.9	2.5 ± 1.7
20%	Baseline	36.5 ± 3.0	94.2 ± 5.4	55.6 ± 2.4	97.0 ± 2.6
	Krum	10.6 ± 0.8	90.5 ± 4.4	24.4 ± 1.5	0.3 ± 0.1
	Trimmed-mean	27.5 ± 3.8	72.8 ± 12.8	34.3 ± 8.1	5.4 ± 4.4
	noise+clip	43.0 ± 1.2	96.5 ± 3.1	31.1 ± 1.1	22.6 ± 6.2
	FLAME	42.4 ± 2.9	77.4 ± 16.1	27.5 ± 2.9	2.7 ± 1.1
40%	Baseline	16.5 ± 5.8	99.3 ± 0.4	58.4 ± 2.2	89.3 ± 1.3
	Krum	10.2 ± 0.3	98.1 ± 1.5	16.6 ± 0.4	2.0 ± 1.5
	Trimmed-mean	14.4 ± 2.9	97.0 ± 2.1	27.8 ± 0.8	29.3 ± 19.2
	noise+clip	33.9 ± 1.4	98.0 ± 1.3	29.5 ± 1.5	29.1 ± 6.1
	FLAME	30.5 ± 2.8	98.3 ± 1.1	29.0 ± 1.8	18.7 ± 8.1

Table 5.3: Comparison of different amount of Malicious clients on various Defense Methods (q=0.9)

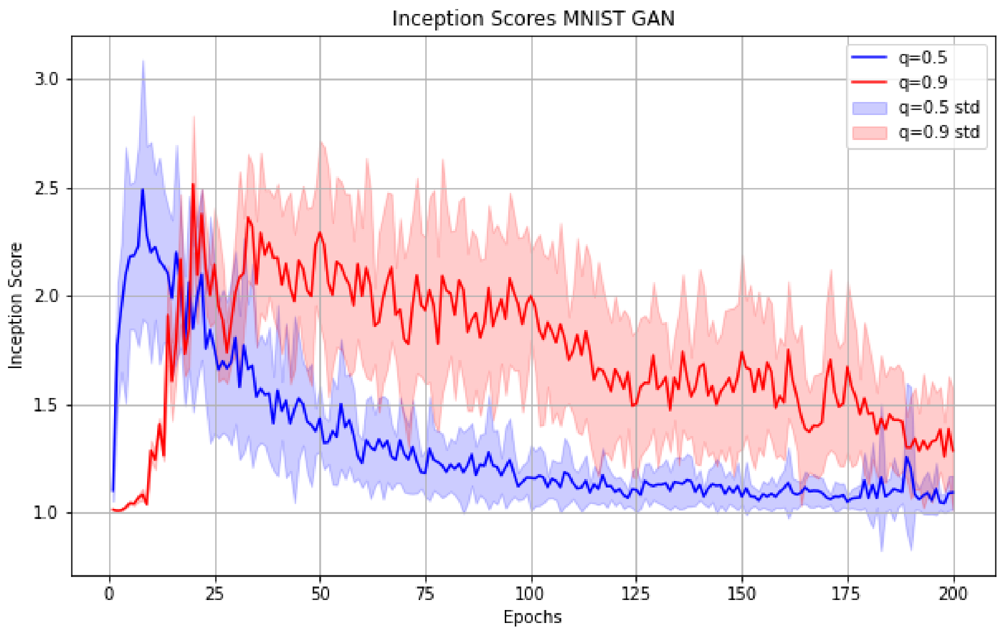

Figure 5.4: Inception Score MNIST

images generally correspond to the targeted label, they exhibit imperfections or vague features that distinguish them from real images.

One possible explanation for this issue is the limited diversity in the dataset used to train the GAN. With insufficient data samples, the GAN may struggle to capture the full range of variations necessary for producing high-quality, realistic images. This lack of diversity can lead to mode collapse or underfitting, where the model fails to generate sufficiently varied or accurate results.
Despite their low quality, the generated samples still lead to improved performance across all tested defense methods. This suggests that as long as the generated samples are somewhat representative of the targeted real samples, they are sufficient for defense mechanisms to effectively identify and filter out poisoned samples from backdoor attacks.

5.2.3. Mitigating Untargeted Backdoor Attacks

This section compares the results of using targeted backdoor attacks with untargeted backdoor attacks. Table 5.4 presents the results of the untargeted attacks. We focus specifically on the highest value of non-IID $q = 0.9$, as this scenario highlights the effects of non-IID and our generated data solution. The comparison shown in Table 5.5, reveals that the results for targeted and untargeted attacks are largely similar. A notable difference is that the MA is higher for untargeted attacks. This can be attributed to the slightly weaker nature of untargeted attacks, which are less effective but pose a lower risk of detection. Despite this, when GANs are utilized, all defense methods, except the noise and clipping method, successfully mitigate the backdoor attacks. The noise and clipping method underperformed because it used the same hyperparameters as in the targeted attack scenario and was not specifically adjusted for untargeted attacks. Combined with the inherently lower impact of untargeted attacks, the defense method of adding noise and clipping would be less effective this way.

5.2.4. Summary

The experimental results highlight the effectiveness of the proposed GAN-based approach in mitigating backdoor attacks across the MNIST, FashionMNIST, and CIFAR-10 datasets. On MNIST and FashionMNIST, incorporating GANs significantly reduced Backdoor Task Accuracy while improving Main Task Accuracy, particularly in highly non-IID scenarios ($q = 0.9$) where conventional defense methods failed without GAN support.

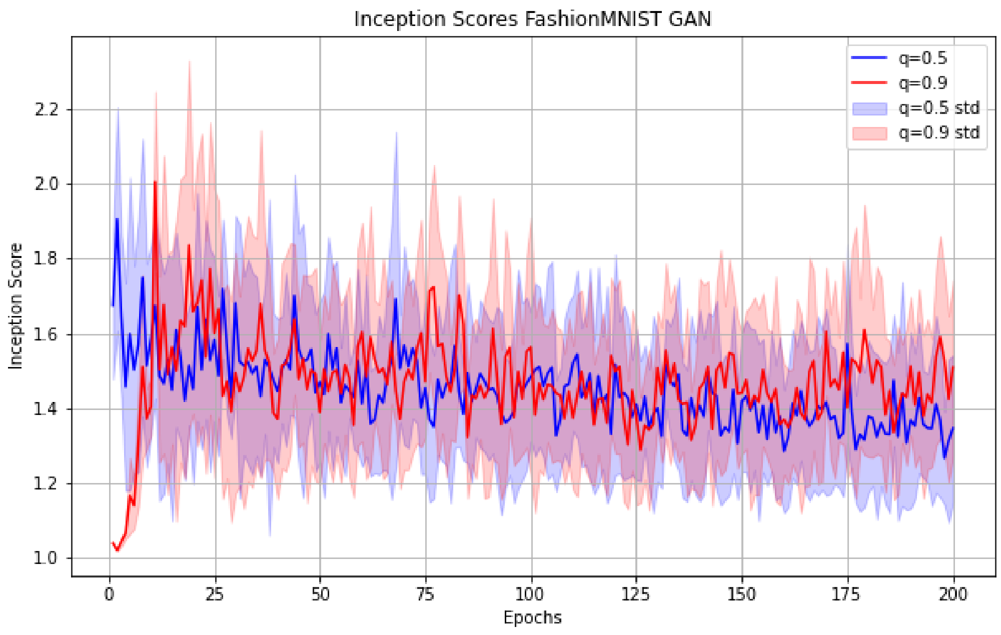

Figure 5.5: Inception Score FashionMNIST

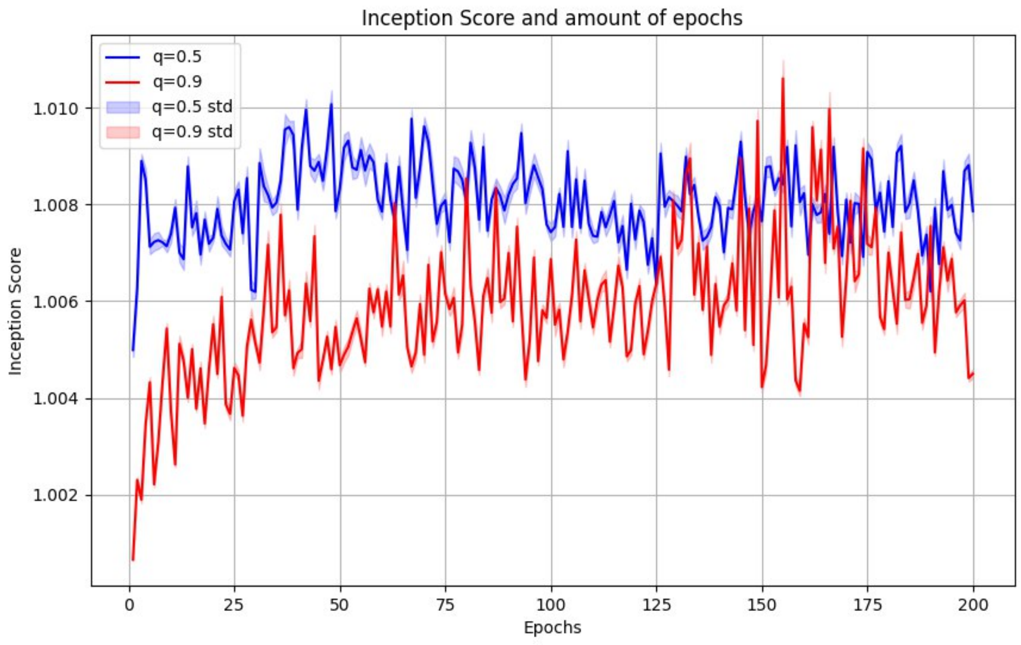

Figure 5.6: Inception Score CIFAR-10

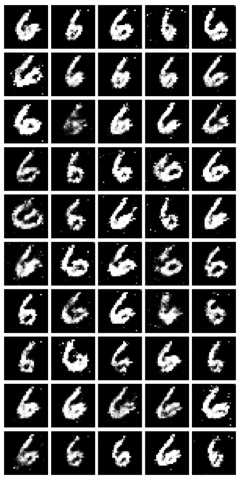
Figure 5.7: Samples of generated images of MNIST label 6

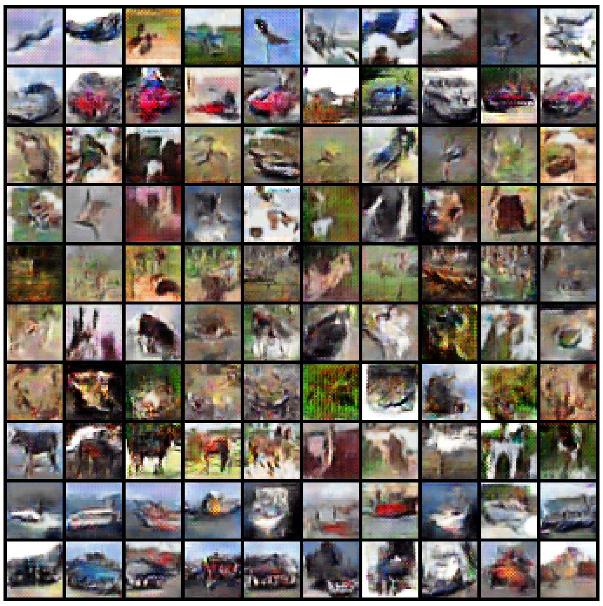
Figure 5.8: Samples of generated images of CIFAR-10 dataset

Scenario	Defense Method	CIFAR-10			
		Without GAN		With GAN	
		MA (%)	BA (%)	MA (%)	BA (%)
IID	Baseline	75.9 ± 0.3	99.8 ± 0.1	-	-
	Krum	72.1 ± 2.0	0.1 ± 0.1	-	-
	Trimmed-mean	74.8 ± 0.3	0.1 ± 0.1	-	-
	noise+clip	62.8 ± 1.6	99.3 ± 0.2	-	-
	FLAME	64.5 ± 1.0	12.5 ± 3.9	-	-
q = 0.5	Baseline	68.8 ± 0.3	99.8 ± 0.1	73.6 ± 1.5	97.1 ± 0.5
	Krum	57.6 ± 1.9	17.1 ± 2.5	27.1 ± 3.5	1.7 ± 1.1
	Trimmed-mean	66.1 ± 1.8	38.6 ± 2.4	67.7 ± 0.5	18.1 ± 0.4
	noise+clip	58.7 ± 2.3	99.4 ± 0.4	42.7 ± 2.0	68.1 ± 8.3
	FLAME	41.4 ± 3.6	27.1 ± 13.5	38.7 ± 0.8	21.7 ± 3.6
q = 0.9	Baseline	38.8 ± 0.7	99.2 ± 0.4	54.5 ± 1.5	95.4 ± 2.0
	Krum	12.9 ± 0.9	98.1 ± 1.1	18.7 ± 4.0	5.3 ± 6.9
	Trimmed-mean	35.6 ± 5.8	54.0 ± 6.1	38.3 ± 0.4	25.0 ± 14.7
	noise+clip	39.1 ± 2.2	98.8 ± 0.1	30.1 ± 2.0	70.5 ± 11.9
	FLAME	35.5 ± 4.1	98.4 ± 1.2	28.9 ± 0.3	16.3 ± 8.7

Table 5.4: Comparison of Main Task and Backdoor Attack Accuracy of various Defense Methods using untargeted attacks on CIFAR-10 (40% malicious of total clients)

Type of Attack	Defense Method	CIFAR-10			
		Without GAN		With GAN	
		MA (%)	BA (%)	MA (%)	BA (%)
Targeted	Baseline	16.5 ± 5.8	99.3 ± 0.4	58.4 ± 2.2	89.3 ± 1.3
	Krum	10.2 ± 0.3	98.1 ± 1.5	16.6 ± 0.4	2.0 ± 1.5
	Trimmed-mean	14.4 ± 2.9	97.0 ± 2.1	27.8 ± 0.8	29.3 ± 19.2
	noise+clip	33.9 ± 1.4	98.0 ± 1.3	29.5 ± 1.5	29.1 ± 6.1
	FLAME	30.5 ± 2.8	98.3 ± 1.1	29.0 ± 1.8	18.7 ± 8.1
Untargeted	Baseline	38.8 ± 0.7	99.2 ± 0.4	54.5 ± 1.5	95.4 ± 2.0
	Krum	12.9 ± 0.9	98.1 ± 1.1	18.7 ± 4.0	5.3 ± 6.9
	Trimmed-mean	35.6 ± 5.8	54.0 ± 6.1	38.3 ± 0.4	25.0 ± 14.7
	noise+clip	39.1 ± 2.2	98.8 ± 0.1	30.1 ± 2.0	70.5 ± 11.9
	FLAME	35.5 ± 4.1	98.4 ± 1.2	28.9 ± 0.3	16.3 ± 8.7

Table 5.5: Comparison Targeted vs Untargeted Backdoor Attacks on CIFAR-10 (40% malicious of total clients) (q = 0.9)

In the more complex CIFAR-10 dataset, GANs also enhanced the performance of various defense methods, as well as under extreme non-IID conditions and when a high proportion of clients were malicious. While GANs consistently reduced BA, slight decreases in MA were observed in scenarios where defense methods were already robust without GANs.

Generated samples for all three datasets exhibited low quality based on Inception Score metrics, with scores well below the threshold for high-quality outputs. Despite this, the samples effectively enhanced the ability of defense mechanisms to identify and mitigate poisoned data, highlighting the utility of GAN-based generated samples even when of suboptimal quality.

Results for targeted and untargeted attacks were broadly similar, with untargeted attacks achieving higher MA due to their inherently weaker and less detectable nature. Defense methods, with the exception of the noise and clipping approach, performed effectively with GANs in mitigating untargeted attacks,

The low quality of generated samples, attributed to limited dataset diversity, suggests that enhancing GAN training could further improve performance. Nonetheless, incorporating GANs into backdoor defense strategies demonstrates substantial improvements, even in challenging conditions.

6 Conclusion

In this thesis, we introduced the concept of federated learning and the challenges it faces in defending against backdoor attacks. While various defense mechanisms have been proposed to mitigate these attacks, they often struggle under non-IID data scenarios. To address this, we proposed a solution based on using GANs for synthetic data generation. This approach aims to reduce the non-IID nature of the local data of clients, thereby enhancing the effectiveness of existing defense mechanisms in non-IID settings.

Our framework simulates a federated learning environment with malicious clients attempting to poison the global model. We evaluate the performance of multiple defense strategies under both IID and non-IID conditions. Subsequently, we apply our GAN-based data generation solution to the non-IID scenarios and assess its impact on defense performance. We measure this using different ratios of non-IID.

Experimental results demonstrate that GANs can effectively mitigate the non-IID characteristics in federated learning by generating data that brings client distributions closer to an IID configuration. Although the quality of the data points generated by the GANs may be relatively low, the results indicate that even low-quality generated data can significantly reduce the non-IID effect. This shift toward an IID-like scenario enables defense mechanisms that otherwise struggle in non-IID settings to more effectively mitigate backdoor attacks. Our findings show that even under a highly non-IID ratio, the GAN-based solution improves the efficacy of defense mechanisms in mitigating both targeted and untargeted backdoor attacks.

There is still potential for further enhancement. Future work could focus on generating higher-quality synthetic data to improve defense performance, employing more advanced GAN architectures and diverse datasets, assessing the computational efficiency of GANs, and exploring the implementation of novel GAN architectures tailored for federated learning clients.

References

[1] Martin Arjovsky and Léon Bottou. "Towards principled methods for training generative adversarial networks". In: *arXiv preprint arXiv:1701.04862* (2017).

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[3] Arjun Nitin Bhagoji et al. "Analyzing federated learning through an adversarial lens". In: *International conference on machine learning*. PMLR. 2019, pp. 634–643.

[4] Peva Blanchard et al. "Machine learning with adversaries: Byzantine tolerant gradient descent". In: *Advances in neural information processing systems* 30 (2017).

[5] Theodora S Brisimi et al. "Federated learning of predictive models from federated electronic health records". In: *International journal of medical informatics* 112 (2018), pp. 59–67.

[6] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. "Density-based clustering based on hierarchical density estimates". In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer. 2013, pp. 160–172.

[7] Mingqing Chen et al. "Federated learning of out-of-vocabulary words". In: *arXiv preprint arXiv:1903.10635* (2019).

[8] Xinyun Chen et al. "Targeted backdoor attacks on deep learning systems using data poisoning". In: *arXiv preprint arXiv:1712.05526* (2017).

[9] Justin Engelmann and Stefan Lessmann. "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning". In: *Expert Systems with Applications* 174 (2021), p. 114582.

[10] Martin Ester et al. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *kdd*. Vol. 96. 34. 1996, pp. 226–231.

[11] Minghong Fang et al. "Local model poisoning attacks to {Byzantine-Robust} federated learning". In: *29th USENIX security symposium (USENIX Security 20)*. 2020, pp. 1605–1622.

[12] Minghong Fang et al. "Local model poisoning attacks to {Byzantine-Robust} federated learning". In: *29th USENIX security symposium (USENIX Security 20)*. 2020, pp. 1605–1622.

[13] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. "Mitigating sybils in federated learning poisoning". In: *arXiv preprint arXiv:1808.04866* (2018).

[14] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. arXiv: `1406.2661 [stat.ML]`. URL: `https://arxiv.org/abs/1406.2661`.

[15] Ishaan Gulrajani et al. "Improved training of wasserstein gans". In: *Advances in neural information processing systems* 30 (2017).

[16] Xu Han, Haoran Yu, and Haisong Gu. "Visual inspection with federated learning". In: *Image Analysis and Recognition: 16th International Conference, ICIAR 2019, Waterloo, ON, Canada, August 27–29, 2019, Proceedings, Part II 16*. Springer. 2019, pp. 52–64.

[17] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[18] Peter Kairouz et al. "Advances and open problems in federated learning". In: *Foundations and trends® in machine learning* 14.1–2 (2021), pp. 1–210.

[19] Diederik P Kingma. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[20] Hatice Kübra Kılınç and O. Fatih Kececioglu. "Generative Artificial Intelligence: A Historical and Future Perspective". In: *Academic Platform Journal of Engineering and Smart Systems* 12 (Feb. 2024). DOI: `10.21541/apjess.1398155`.

[21] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[22] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[23] Tian Li et al. "Federated learning: Challenges, methods, and future directions". In: *IEEE signal processing magazine* 37.3 (2020), pp. 50–60.

[24] Zijian Li et al. "Federated learning with gan-based data synthesis for non-iid clients". In: *International Workshop on Trustworthy Federated Learning*. Springer. 2022, pp. 17–32.

[25] Yingqi Liu et al. "Trojaning attack on neural networks". In: (2017).

[26] Brendan McMahan et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[27] Mehdi Mirza. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[28] Thien Duc Nguyen et al. "FLAME: Taming Backdoors in Federated Learning". In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1415–1432. ISBN: 978-1-939133-31-1. URL: `https://www.usenix.org/conference/usenixsecurity22/presentation/nguyen`.

[29] V Chawla Nitesh. "SMOTE: synthetic minority over-sampling technique". In: *J Artif Intell Res* 16.1 (2002), p. 321.

[30] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Yulia R Gel. "Defending against backdoors in federated learning with robust learning rate". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 10. 2021, pp. 9268–9276.

[31] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *CoRR* abs/1912.01703 (2019). arXiv: `1912.01703`. URL: `http://arxiv.org/abs/1912.01703`.

[32] Alec Radford. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[33] Alec Radford. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[34] Alec Radford. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[35] Nicola Rieke et al. "The future of digital health with federated learning". In: *NPJ digital medicine* 3.1 (2020), pp. 1–7.

[36] Tim Salimans et al. "Improved techniques for training gans". In: *Advances in neural information processing systems* 29 (2016).

[37] Tim Salimans et al. "Improved techniques for training gans". In: *Advances in neural information processing systems* 29 (2016).

[38] Ziteng Sun et al. "Can you really backdoor federated learning?" In: *arXiv preprint arXiv:1911.07963* (2019).

[39] Chen Wu et al. "Mitigating backdoor attacks in federated learning". In: *arXiv preprint arXiv:2011.01767* (2020).

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

[41] Chulin Xie et al. "Dba: Distributed backdoor attacks against federated learning". In: *International conference on learning representations*. 2019.

[42] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. "Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation". In: *Uncertainty in Artificial Intelligence*. PMLR. 2020, pp. 261–270.

[43] Wensi Yang et al. "Ffd: A federated learning based method for credit card fraud detection". In: *Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8*. Springer. 2019, pp. 18–32.

[44] Dong Yin et al. "Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, Oct. 2018, pp. 5650–5659. URL: `https://proceedings.mlr.press/v80/yin18a.html`.

[45] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. "Time-series generative adversarial networks". In: *Advances in neural information processing systems* 32 (2019).

[46] Han Zhang et al. "Self-attention generative adversarial networks". In: *International conference on machine learning*. PMLR. 2019, pp. 7354–7363.

[47] Yue Zhao et al. "Federated learning with non-iid data". In: *arXiv preprint arXiv:1806.00582* (2018).