Securing Federated Learning: A Comprehensive Defence Against Privacy Attacks

M A Moyeen

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Electrical and Computer Engineering) at

Concordia University

Montréal, Québec, Canada

July 2025

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. M A Moyeen**

Entitled: **Securing Federated Learning: A Comprehensive Defence Against Privacy Attacks**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Electrical and Computer Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Chunjiang An

_____ External Examiner
Dr. Biplab Sikdar

_____ Arm's Length Examiner
Dr. Mohammad Reza Soleymani

_____ Examiner
Dr. Tristan Glatard

_____ Examiner
Dr. Yan Liu

_____ Thesis Supervisor
Dr. Anjali Agarwal

_____ Thesis Co-supervisor
Dr. Kuljeet Kaur

Approved by _____
Dr. Jun Cai, Graduate Program Director
Department of Electrical and Computer Engineering

_____ 2025 _____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Securing Federated Learning: A Comprehensive Defence Against Privacy Attacks

M A Moyeen, Ph.D.

Concordia University, 2025

In this information age, machine learning (ML) applications drive smart living through innovations such as personalized healthcare, intelligent transportation, and smart homes. However, despite these advancements, businesses and industries continue to face significant challenges in safeguarding data privacy, as ML systems increasingly rely on vast amounts of user data. In this direction, Federated Learning (FL) has emerged as a promising solution, enabling collaborative model training while keeping user data on local premises without the need to share raw data. However, FL faces significant challenges also, including expensive communication costs, system heterogeneity, and vulnerability to various attacks. In particular, it is susceptible to poisoning attacks, where malicious participants corrupt models and data as well as inference attacks that exploit gradients to reveal sensitive information through membership inference or model inversion techniques. These attacks can extract sensitive information from shared gradients, undermining the fundamental privacy guarantees of federated systems. Thus, effective defence mechanisms are fundamental for fully leveraging the advantages of FL. Numerous defences, such as FoolsGold, Flod, Flad, MADDPG, and others, are in place to secure the FL systems.

However, the majority of the defence mechanisms suffer from accuracy degradation, computational overhead, and inadequate attack prevention. Most client selection methods cannot reliably separate malicious and straggler clients, with even cutting-edge approaches struggling with herding and cold-start issues. Furthermore, recent state-of-the-art techniques frequently fail to defend against inference attacks adequately. These methods typically employ Secure Multiparty Computation (SMPC), Homomorphic Encryption (HE), or Differential Privacy (DP) as defensive measures.

However, SMPC and HE suffer from high computational complexity, while DP often leads to degraded model accuracy.

Thus, this research addresses these limitations by proposing five different defence mechanisms that ensure robust FL with protected gradients. The proposed mechanisms consist of *FedChallenger*, *Fed-Reputed*, *SignDefence*, *Ada-Sign*, and *SignMPC*. The proposed *FedChallenger* introduces a dual-layer defence mechanism that comprises the zero-trust challenge-response-based authentication at the first layer and a variant of Trimmed-Mean aggregation at the second layer that leverages pairwise cosine similarity and Median Absolute Deviation (MAD). Extensive evaluation on MNIST, FMNIST, EMNIST, and CIFAR-10 datasets demonstrates 3-10% accuracy improvement over state-of-the-art approaches with 1.1-2.2 times faster convergence and 2-3% higher F1-scores.

Subsequently, the reputation-based client selection approach, *Fed-Reputed*, leverages device capability information and a modified Bellman equation within a hierarchical framework, integrated into a Deep Q-Learning Network (DQN)-based Imbalanced Classification Markov Decision Process (ICMDP) classifier for enhanced client selection. Testing on MNIST and FMNIST datasets demonstrates 9-50% accuracy gains and 1.3-1.7 times faster convergence while effectively detecting both malicious and straggler clients.

Moreover, existing methods often suffer from the dying ReLU problem, where neurons permanently deactivate during training. To counter the dying ReLU problem, *SignDefence* implements a sophisticated aggregation scheme that utilizes sign direction and LeakyReLU-based aggregation, incorporating Jaccard similarity derived from binary-encoded model weights. This technique demonstrates consistent accuracy and F1-score improvements across different attack conditions.

Despite its benefits, *SignDefence* remains vulnerable to inference attacks and suffers from limited generalization due to its fixed threshold across diverse benchmark datasets. To address these limitations, a lightweight strategy, *Ada-Sign*, employs adaptive threshold computation and incorporates DP mechanisms. This approach maintains comparable accuracy to *SignDefence* while providing enhanced gradient protection through adaptive DP. Extensive evaluation on MNIST and HAR datasets reveals 3-20% accuracy improvement for *Ada-Sign* over the majority of state-of-the-art techniques.

Finally, to enhance the protection of both *SignDefence* and *Ada-Sign* against inference attacks,

SignMPC integrates highly configurable SMPC, HE, and DP algorithms. This combined approach ensures comprehensive communication security and gradient privacy while avoiding significant performance bottlenecks. Comprehensive evaluation on MNIST and HAR datasets demonstrates 4-17% accuracy gains for *SignMPC* over established approaches while maintaining computational efficiency and robust privacy guarantees.

Keywords: *Differential Privacy, Federated Learning, Homomorphic Encryption, Inference Attacks, Machine Learning, Poisoning Attacks, Robust Aggregation, Secure Multi-Party Computation.*

Acknowledgments

The completion of this PhD thesis represents a journey of perseverance and growth that would not have been possible without the extraordinary support of many individuals.

I extend my profound gratitude to my supervisors, Dr. Anjali Agarwal and Dr. Kuljeet Kaur, whose exceptional mentorship and unwavering support have been instrumental in my academic success. Both demonstrated remarkable flexibility, offering academic guidance and tremendous mental support throughout this journey. Their patience, dedication, and continuous encouragement have significantly enriched this research.

My heartfelt appreciation goes to the members of my thesis advisory committee, whose thoughtful suggestions and scholarly contributions have strengthened this work immeasurably.

I gratefully acknowledge the NSERC Alliance grant supporting this work in collaboration with Cistech Ltd., Ottawa, Canada.

I owe an immeasurable debt of gratitude to my beloved parents, son, and wife, whose unconditional love, support, and sacrifices have been the foundation upon which all my achievements rest. Their constant unwavering encouragement have been a source of strength throughout this journey.

This achievement stands as a testament to the collective support, love, and belief of everyone who has been part of this transformative chapter of my life. I am profoundly humbled and grateful to each person who contributed to making this dream a reality.

Contents

List of Figures

List of Tables

List of Acronyms

- **AI**: Artificial Intelligence
- **CCPA**: California Consumer Privacy Act
- **CRS**: Client Reputation Systems
- **DP**: Differential Privacy
- **DQN**: Deep Q-Learning Network
- **DRL**: Deep Reinforcement Learning
- **FAT**: Federated Adversarial Training
- **FedDynAT**:Federated Adversarial Training with Dynamic Weighting
- **FedAvg**: Federated Averaging
- **FL**: Federated Learning
- **FAL**: Federated Adversarial Learning
- **GDPR**: General Data Protection Regulation
- **GBM**: Gradient Boosting Machine
- **HDBSCAN**: Hierarchical Density-Based Spatial Clustering of Applications with Noise
- **HE**: Homomorphic Encryption
- **HFL** Hierarchical Federated Learning.
- **ICMDP**: Imbalanced Classification Markov Decision Process
- **IID**: Independent and Identically Distributed
- **IoT**: Internet of Things
- **LR**: Linear Regression
- **MAD**: Median Absolute Deviation
- **MADDPG**: Multi-Agent Deep Deterministic Policy Gradient
- **ML**: Machine Learning
- **MLP**: Multilayer Perceptron
- **MK-FHE**: Multi-Key Fully Homomorphic Encryption
- **MVFLS** Multi-participant Vertical FL based on Secret Sharing
- **NLP**: Natural Language Processing
- **NTRU** Nth degree TRuncated polynomial ring Unit
- **ReLU**: Rectified Linear Unit

- **RS**: Random Selection
- **SCS**: Stochastic Client Selection
- **SecAgg**: Secure Aggregation
- **SGD**: Stochastic Gradient Descent
- **SHAP**: Shapley Additive exPlanations
- **SIP**: Stochastic Integer Program
- **SMPC**: Secure Multi-Party Computation
- **SPCC**: Secure and Private Communication Channels
- **VFL**: Vertical Federated Learning
- **ZB**: Zettabytes

Chapter 1

Introduction

The expeditious evolution of Artificial Intelligence (*AI*) has blessed us with numerous smart and life-easing applications, creating a decent-sized global AI market valued at 136.55 billion USD in 2022 [105]. However, the demand for intelligent AI applications continues to increase day by day due to the need for intelligent applications that can learn, adapt, and produce new concepts and results. According to [87], the AI market is expected to expand at a 37.3% annual growth rate between 2023 and 2030. However, the core component of AI is the Machine Learning (*ML*) algorithm, which builds statistical models through the accumulation of knowledge using training data. In this process, the ML algorithm initially learns the mapping between input data and output data on a centralized server. After multiple rounds of training and verification, the model is prepared for prediction on unseen data samples. Therefore, the model requires a substantial amount of data for accurate predictions.

This demand for data aligns with the proliferation of the Internet of Things (*IoT*) devices, which has led to exponential growth in data generation. Specifically, IoT devices are projected to produce more than 600 zettabytes (ZB) of data by 2030 [97], driven by the expected rise of interconnected devices to 500 billion by then [6]. This growth has created new opportunities for ML to extract insights from the data and enable intelligent decision-making.

However, the privacy concerns including including potential data breaches, unauthorized access, and the misuse of confidential information of edge IoT devices often become a barrier to the evolution of smart applications. Federated Learning (*FL*) has shown promise in addressing these privacy

concerns due to its architectural benefits. In vanilla FL, a central server selects random clients that are edge devices wishing to participate in the federated training process and initiates them with the base model. These clients train the base model with their respective local data without revealing it outside their periphery. Furthermore, frequent updates of client-server model parameters and server-side aggregation enhance the model's accuracy during training. As a result, federated training not only improves efficiency but also serves as a privacy-preserving alternative to traditional machine learning solutions.

1.1 Motivation

FL appears to be a promising type of ML training architecture as it ensures zero movement of sensitive data from client machines. However, the benefits of privacy-preserving FL are limited by various types of issues and security attacks, namely, poisoning attacks, inference attacks, backdoor attacks, device heterogeneity, and extensive communication overhead [73, 112]. Amongst these issues, poisoning attacks, a type of Byzantine attack [64], and inference attacks are considered severe security issues that hinder the adoption of FL systems since these attacks can compromise model accuracy and data privacy. Thus, The primary focus of this research is on poisoning attacks and inference attacks. The former can be categorized into model poisoning and data poisoning attacks. Model poisoning attacks involve an attacker intentionally manipulating the model parameters during ML training to introduce biases or inaccuracies that will ultimately compromise the model's ability to make accurate predictions [131]. On the other hand, data poisoning attacks involve an attacker manipulating the data used to train an ML model to cause the model to make incorrect predictions [54]. This can be accomplished by adding intentionally mislabelled data or by manipulating existing data features to change their meaning. Additionally, this research focuses on inference attacks [106], which exploit various techniques to deduce sensitive information from seemingly innocuous data, including membership inference and model inversion attacks [60]. These attacks aim to uncover private details about individuals or the training data used in ML models.

The research suggests that the first line of defence against poisoning attacks is to select non-malicious clients [39]. Trusted client selection may eliminate the existence of malicious updates and

prevent attack propagation to the FL aggregation process. In this direction, client reputation-based systems [108] can ensure the selection of trusted clients; however, they are limited to classification challenges between trusted and untrusted participants. Furthermore, determining the reputation score is difficult to define at the initial stages of communication.

Despite proper client selection, privacy breaches and model poisoning can occur through unprotected communication, potentially undermining the FL benefit. Thus, in order to circumvent these shortcomings, Secure Multi-Party Computation (*SMPC*) strategies are often found to be helpful in defending poisoning attacks as a second line of defence because they do not reveal individual model parameters [19]. In particular, these techniques may integrate Homomorphic Encryption (*HE*) or Differential Privacy (*DP*) while performing secure aggregation. However, these approaches have significant performance bottlenecks and are often found to be impractical to use and are mostly not scalable [39].

Therefore, a third line of defence mechanism may incorporate an efficient server-side aggregation mechanism to remove the impact of malicious updates in the model parameter aggregation process. Thus, researchers have proposed various central server-side aggregation mechanisms including Krum [15], FedAvg [79], Trimmed-Mean [143], and Fang *et al.'s* approach [34]. Krum selects the k clients with the smallest Euclidean distance from the median of all model updates from all clients and aggregates their updates. In contrast, the Trimmed-Mean calculates pairwise Euclidean distance but removes model updates that exceed the median of model updates. Meanwhile, Fang *et al.'s* approach removes outliers by analysing the loss impact of model parameters. However, FedAvg, the baseline approach of FL, does not remove any outliers; instead, it aggregates model parameters with the Federated Averaging (FedAvg) algorithm. Despite the benefits of aggregation strategies, they are only suitable when extreme values skew the results.

For stronger security, Byzantine fault-tolerant algorithms with model verifications [64] can handle malicious client updates propagated to a central server for aggregation. Thus, the aggregation mechanism does not need to be concerned about malicious aggregation. However, these algorithms are vulnerable to collusion attacks. Unlike traditional Byzantine fault-tolerant methods that assume independent malicious actors and use statistical filtering, colluding attackers coordinate their model updates to appear consistent. This coordinated manipulation can poison the global model, degrading

performance or introducing backdoors [139].

Moreover, aggregation mechanisms often run out of samples due to the removal of malicious updates. Therefore, the FL server has to restart the training process, which incurs substantial communication and computational overhead [152]. Hence, to alleviate this problem, a predictive data recovery strategy can regenerate the data points to meet the required sample size for aggregation. However, existing regeneration strategies are mainly based on optimization strategies and consider a pre-trained local model. Thus, they incur more computational overhead and loss of accuracy [29, 155]. Although, the scheme proposed in [32], named as Flod overcomes the majority of the data clipping problems with sign-magnitude and Hamming distance-based τ clipping. However, it still struggles with increased data noise and the dying ReLU problem [146].

Indeed, hybrid defence strategies often combine multiple security and privacy techniques for better privacy protection. However, recent hybrid defence techniques such as Stake [31], Shapley-based methods [56], and Cluster-based approaches [50] each employ distinct strategies against poisoning attacks but face two significant challenges: high computational costs and reduced accuracy with non-IID data distributions. For conciseness, we refer to these methods as Stake, Shap, and Cluster hereafter. The Stake method utilizes blockchain technology for majority voting, along with client reward and penalty calculations, which enhances security but also introduces significant processing delays. Shap relies on SHAP value computations, creating substantial computational demands, especially for complex models. Meanwhile, the Cluster technique analyses gradients from client updates to identify source and target classes before applying Hierarchical Density-Based Spatial Clustering of Applications with Noise (*HDBSCAN*) [142] clustering to detect malicious updates. This approach proves particularly sensitive to variations in data distribution.

In summary, current FL defence mechanisms against poisoning and inference attacks are limited by high computational costs, reduced accuracy with non-IID data, and vulnerabilities to sophisticated collusion. Thus, there's a critical need for an efficient and robust defence strategy that can overcome these practical challenges while maintaining high security and performance.

1.2 Problem Statement

Despite existing defences, a critical challenge in FL is developing poisoning and inference attack-resistant systems that integrate robust client selection and SMPC, and importantly, can efficiently recover from malicious disruptions with minimal computational and communication overhead. [49]. In this direction, the literature suggests that the existing defence mechanisms in FL attempt to protect against model or data poisoning attacks using verification-based rejection [9, 154] or robust aggregation [15, 143, 34] strategies. However, to prevent the propagation of compromised samples in global aggregation, it is necessary to detect and remove compromised devices and their respective updates before they affect the performance of the trained model. Moreover, devices do not know if they are interacting with compromised devices and/or aggregating compromised model updates, and existing verification-based defence mechanisms for models suffer mainly from accuracy and performance bottlenecks [131]. The existing approaches based on aggregation strategies require adjusting their approach based on data type and dimensions. More importantly, most of them cannot provide alerts when the affected sample does not skew the results [34].

Beyond these concerns, the selection of clients from existing algorithms is mainly based on random sampling [27] or weighted performance feedback [47] or reinforcement learning [149], increasing the chance of including malicious clients [115]. Some reputation-based scoring mechanisms have attempted to alleviate this problem [124], but, they are subject to the correctness of reputation scores; which in turn might be altered by the malicious actors. Moreover, collusion attacks make the situation even worse, as multiple malicious clients can alter the system's behaviour.

In addition, the communication channel between clients and servers remains a critical point for attackers. Attackers can breach the information propagating through these channels and deviate model parameters. Therefore, a DP approach [118] has been proposed in the literature; which ensures that data remains unaltered over the open communication channel. However, DP mechanisms are not fully secure and degrade model accuracy [126]. Thus, it requires special encryption strategies without degrading system performance. Even though HE solutions are found to be useful in alleviating the problem of insecure communication related to privacy leakage, they incur huge computational overhead and require a significant amount of information exchange [148]. Moreover,

other strategies, such as secure aggregation or trust-based computation, cannot fully guarantee efficiency; instead, they degrade the model's accuracy [131]. Furthermore, aggregation mechanisms may run out of samples for accumulation because of their outlier removal strategy and can trigger repeated training rounds [29]. Also, state-of-the-art techniques [32] that employ gradient signing for poisoning attack mitigation struggle with the dying ReLU problem and often lose accuracy due to the choice of the wrong gradient direction. Additionally, recent hybrid methods [31, 56, 50] may advance poisoning detection, but their practical deployment remains constrained by computational and data distribution limitations in real-world FL scenarios.

Since state-of-the-art techniques do not guarantee a poisoning-free environment, a robust defence mechanism capable of detecting, preventing, and efficiently recovering from such attacks remains critical.

1.3 Goals and Objectives

In this research work, we consider a multilayer defence mechanism against poisoning and inference attacks to ensure the data privacy and integrity of federated systems. The overall objective is to fully defend against and recover from poisoning and inference attacks irrespective of the dataset and training environment. This research has the following three main objectives.

- Develop a highly efficient, robust aggregation strategy to recover from poisoning attacks irrespective of datasets and training environments.
- Devise an adaptive client reputation system that can predict client reputation scores for device selection and client update selection.
- Design an efficient SMPC algorithm to defend against inference attacks.

The overall goal is to design and evaluate an efficient and robust defence mechanism that can detect compromised devices, and then discard their malicious updates. Overall, it focuses on defending against poisoning and inference attacks without incurring excessive overhead. We will discuss these concepts in more detail in subsequent chapters.

1.4 Thesis Contributions

The main contribution of this thesis is to design and develop a secure defence mechanism that can defend against poisoning attacks and recover from any attack impacts. The defence mechanism involves assessing device reputations and devising reliable client selection approach. It also incorporates SMPC systems to protect the open communication. The recovery phase aims to develop dataset and training environment-independent aggregation strategies to remove malicious updates or reconstruct benign updates from malicious ones. The reconstruction ensures that model training does not suffer from a lack of data samples. In summary, the contributions of this thesis comprise of three connected components: robust aggregation, client selection, and inference attack-resistant techniques.

1.4.1 Robust Aggregation

The robust aggregation mechanism defends against poisoning attacks by mitigating the impact of malicious client updates during model aggregation. To achieve this, this thesis component proposes *FedChallenger*, which not only serves as an aggregation strategy but also helps in restricting malicious participants using a challenge-response mechanism and cosine-similarity-based filtering. The proposed technique has two versions. The former version incorporates fixed threshold-based cosine similarity filtering [82]. The latter version introduced a revised zero-trust challenge-response architecture [88] that actively authenticates all participating devices before and during training sessions. The framework incorporates an improved robust aggregation algorithm utilizing Median Absolute Deviation (MAD)-based [46] trimming to enhance resilience against poisoning attacks. Furthermore, the performance of the extended version is evaluated across multiple benchmark datasets [135], including comparisons with the most recent techniques [31, 56, 50].

The major contributions of this thesis component are as follows.

- Propose a zero-trust challenge-response-based defence mechanism named *FedChallenger* to detect and prevent poisoning attacks.
- Present a trust-based attack detection algorithm that relies on challenge-response information to compute trust.

- Introduce a robust aggregation mechanism that applies cosine similarity-based consensus boosting to benign weights and dynamically prunes malicious updates via MAD for adversarial updates.
- Evaluate the performance of the proposed approach on MNIST, FMNIST, EMNIST, and CIFAR-10 datasets [135] using different evaluation metrics such as convergence time and accuracy.

Often challenge-response information accumulation may not be feasible due to deployment environment or communication burden. Therefore, we proposed another lightweight alternative to *FedChallenger* using sign-magnitude of gradients called *SignDefence* [83]. The major contributions of this thesis component are highlighted as follows:

- Design the *SignDefence* technique that uses sign magnitude of gradients and LeakyReLU with Jaccard Similarity for weight estimation to effectively mitigate poisoning attacks.
- Present an FL architecture that utilizes *SignDefence* aggregation strategies.
- Propose the *SignDefence* algorithm to train the model with no or minimal impact of poisoning samples.
- Evaluate the performance of the proposed *SignDefence* technique using the Human Activity Recognition (HAR) dataset [41] with current state-of-the-art Krum [15], Trimmed-Mean [143], FedAvg [79], Flame [86], and Flod [32] techniques. Further, the evaluations are rigorously conducted under no-attack, attack on Trimmed-Mean, attack on Krum, Min-Max attack, Min-Sum attack, and Label Flipping attack scenarios.

1.4.2 Client Selection

This component of the thesis introduces *Fed-Reputed*, a Deep Q-Learning Network (DQN)-based reputation-driven client selection framework tailored explicitly for HFL in biomedical devices. It is a Deep Reinforcement Learning (DRL)-based HFL strategy that utilizes DQN with Imbalanced Classification Markov Decision Process (ICMDP) [65] for the unbiased selection of edge servers and clients, thereby alleviating the rising privacy concerns. More specifically, the ICMDP

utilizes device capability information to overcome the cold-start problem. The overall contributions of this thesis component can be summarized as follows:

- Propose *Fed-Reputed*, a strategy inspired by state-of-the-art DQN strategies and ICMDP to select proper clients during cold-start and regular phases of training.
- Devise an HFL architecture that leverages the benefits of ICMDP to classify and penalise malicious and straggler nodes efficiently.
- Propose a modified Bellman equation by incorporating the reputation score, which is computed by the historical Q value, straggler status, malicious status, model training performance, and resource usage, respectively.
- Extensively evaluate *Fed-Reputed* on the MNIST and FMNIST datasets and compare it with SCS [113], MADDPG [77], and RS [79] approaches under different straggler and malicious scenarios.

1.4.3 Inference Attack-Resistant Techniques

To ensure inference attack resistance, this thesis component introduces *Ada-Sign* and *SignMPC*. *Ada-Sign* is an adaptive sign-based aggregation method that significantly enhances the robustness of FL against poisoning attacks and data heterogeneity. *Ada-Sign* builds upon the concept of signed gradients but introduces several key innovations:

- Introduce a dynamic thresholding mechanism that automatically adjusts the contribution weight of each gradient dimension based on Jaccard similarity of gradient signs and MAD computation, eliminating the need for manual threshold tuning of *SignDefence*.
- Propose a novel soft weighting scheme that replaces *SignDefence*'s binary weighted average with continuous weights, reducing information loss from false positives. Also, Leaky Weighted aggregation prevents the dying ReLU problem effectively like LeakyReLU with stable accuracy improvement.
- Devise a DP robust aggregation algorithm to protect gradients against inference and reconstruction attacks.

- Introduce a Jaccard-based reliability metric that detects subtle sign-flipping attacks effectively and retains *SignDefence*'s computational efficiency while protecting from inference attacks.

By incorporating these advancements, *Ada-Sign* provides a more resilient, adaptive, and efficient defence mechanism for FL, ensuring the integrity and performance of the global model even in the presence of sophisticated adversaries and diverse data distributions. Moreover, it is resilient against inference and reconstruction attack types due to the involvement of DP.

To further enhance the privacy and security of model updates, *SignMPC* incorporates Secured Multi-Party Computation (SMPC). It leverages a multi-layered defence strategy, integrating state-of-the-art privacy-enhancing techniques with an innovative robust aggregation mechanism. The framework's core novelty lies in its ability to adaptively aggregate client updates based on directional consensus, even when those updates have been perturbed for privacy and transmitted securely. The key contributions of this work are summarized as follows:

- A unified, multi-layered FL aggregation framework, *SignMPC*, that synergistically combines DP, SMPC, and a novel robust aggregation strategy to offer comprehensive protection against both privacy breaches and poisoning attacks.

- An optimized robust aggregation algorithm that dynamically weights client contributions based on an adaptive Jaccard similarity threshold of their gradient signs. This approach offers superior resilience to data poisoning and model divergence attacks compared to traditional methods.

- The seamless integration of client-side DP to perturb individual gradient updates, ensuring privacy guarantees for client data before transmission.

- The incorporation of SMPC principles to facilitate the secure transmission and aggregation of client updates, preventing the central server from observing raw, individual gradient contributions even after DP perturbation.

- A detailed architectural design for *SignMPC*, elucidating the functional modules on both the client and server sides, and illustrating the precise data flow through each privacy and robustness enhancement stage.

1.5 Organization

The remainder of this thesis is organized as follows. In the next chapter, the relevant background is presented. In Chapter 3, the literature review is discussed. Chapter 4 illustrates the first defence mechanism, i.e., *FedChallenger*. Chapter 5 introduces our second mechanism named *SignDefence*. The details for the third mechanism namely, *Fed-Reputed* is provided in Chapter 6. Chapter 7 deatils *Ada-Sign*. Following this, Chapter 8 describes the proposed *SignMPC* technique. Finally, Chapter 9 concludes the thesis with future research directions.

Chapter 2

Background

In this chapter, we present the relevant background to this thesis. In particular, this chapter discusses the following topics: classical ML, FL, Byzantine Attacks, Robust Aggregation, Client Selection, Secure and Private Communication Channels, Adversarial Training, Model Parameter Recovery, and Byzantine Attack Defence Framework.

2.1 Machine Learning

Classical ML algorithms are designed to find patterns in the data and make predictions based on those patterns. These algorithms were initially based on statistics and probabilistic reasoning by measuring the distance between data points, vectors, directions, values, intensities, etc.

Supervised ML and unsupervised ML are two main branches of classical ML. In supervised learning, supervision is used to show the algorithm the correct output label. This data is labeled, hence the name supervised learning. On the other hand, in unsupervised learning, the machine is left to figure out the structure of the data it feeds. This data is not labeled, nor does it have a teacher, and therefore is called unsupervised learning.

However, classical ML involves centralized data training, where the data is gathered, and the entire training process executes at the central server. Therefore, data leaves the user machine in the untrusted world and raises serious privacy concerns. In Fig. 2.1, the classical ML architecture is shown [66]. The architecture suggests that the data moves from the client to the central server.

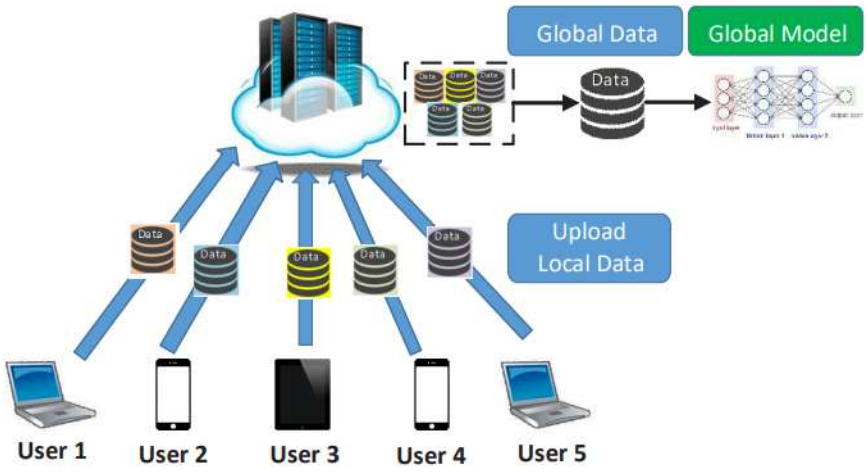


Figure 2.1: Classical ML Training [66].

2.2 Federated Learning

FL is an ML approach that allows multiple devices or entities (such as smartphones, IoT devices, or edge servers) to collaboratively train a model while keeping the data decentralized and at its source. This decentralized approach has numerous advantages and applications in various domains.

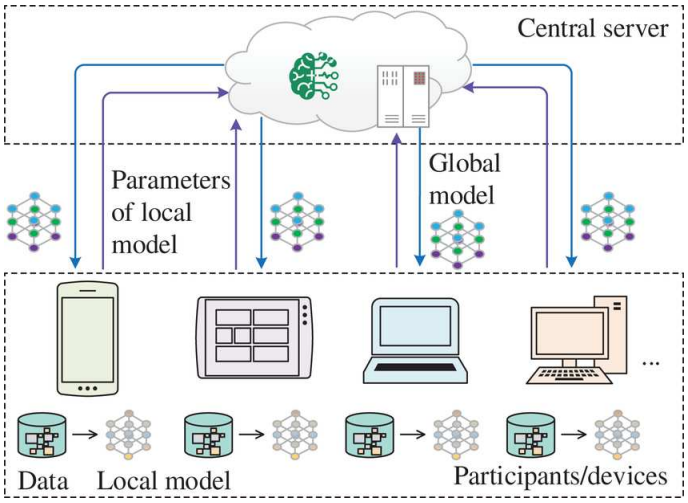


Figure 2.2: FL Training [144].

One of the key benefits of FL is privacy preservation. Instead of uploading raw data to a central

server, FL trains a model locally on each device, using data that remains on the device. Only the model updates (such as gradients) are shared and aggregated across devices. This ensures that sensitive data, such as personal information or proprietary business data, doesn't leave the user's device or organization's infrastructure, reducing the risk of data breaches or leaks. Fig. 2.2 shows a sample FL architecture wherein it is evident that the client does not send their data to the server.

Another advantage of FL is its ability to handle large-scale distributed data. With the proliferation of edge devices and IoT devices that generate large amounts of data, FL allows these decentralized devices to collaborate and collectively learn from the data without the need to centralize and transfer the data to a central server. This significantly reduces the bandwidth and computational requirements, making FL feasible for resource-constrained devices.

FL also enables personalized and context-aware models. Since the training occurs locally on individual devices, models can be tailored to the specific user's preferences and behaviour. For example, in the case of predictive text on a smartphone keyboard, the model can learn the user's typing patterns without needing to know the exact text being typed. Additionally, FL can capture the context of individual devices, such as location, time, or user interactions, without compromising user privacy.

In terms of applications, FL has numerous use cases. One prevalent application is in healthcare [2], where sensitive patient data can be kept secure while enabling collaborative model training across hospitals or research institutions. This allows for the development of robust predictive models while complying with strict privacy regulations. Also, FL has applications in industries like finance [76], where financial institutions can collaboratively train models on customer data to detect fraud patterns without sharing sensitive customer information. Furthermore, FL is helpful in scenarios where internet connectivity is limited or unreliable, such as remote areas or onboard vehicles. In these cases, devices can locally train a model and periodically synchronize with a central server when connectivity is available [152].

Beyond these, FL is increasingly applied in smart city initiatives, allowing various urban sensors and devices to collectively improve services like traffic management, energy optimization, or public safety without centralizing all raw data [85]. In the realm of personalized user experiences,

mobile devices can leverage FL to learn user preferences (e.g., keyboard prediction, recommendation systems) directly on the device, enhancing privacy while improving model accuracy. Moreover, FL is being explored in industrial IoT for predictive maintenance, where machinery from different factories can collaboratively train models to predict equipment failures without sharing proprietary operational data [152]. Lastly, in edge computing, FL enables distributed intelligence, allowing devices at the network edge to process and learn from data locally, reducing latency and bandwidth consumption while improving overall system responsiveness [123].

2.3 Byzantine Attacks

The byzantine attack poses a significant challenge in FL because it breaks the assumption that participating clients are honest and follow the predefined protocol. Data poisoning and model poisoning are both considered Byzantine attacks in FL. In data poisoning attacks, malicious clients upload poisoned data to the global model, which can lead to a significant drop in the model's accuracy. Fig. 2.3 represents a data poisoning attack where label two is trained as label seven. Thus, the poisoned model outputs label seven instead of two.

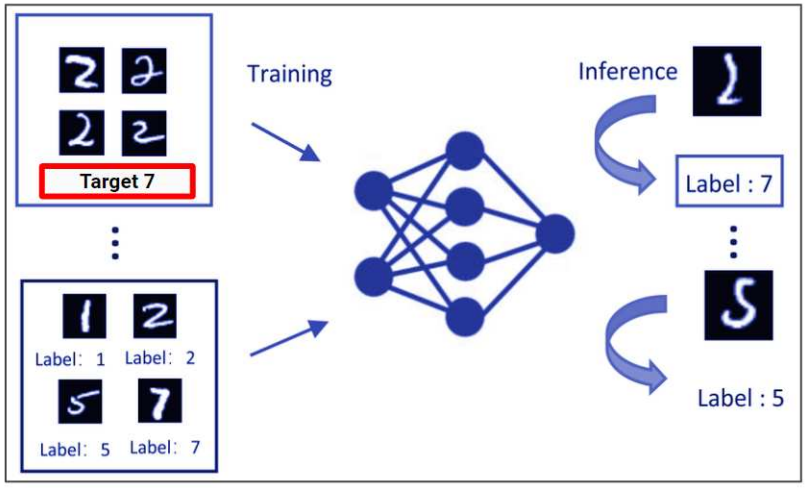


Figure 2.3: Data Poisoning (Label Flipping) Attack [92].

In model poisoning attacks, malicious clients upload poisoned models to the global model, which can also lead to a significant drop in the model's accuracy. Fig. 2.4, which has been modified

from [89], shows that the adversary adds up malicious updates, and that propagates in federated averaging. Therefore, the poisoned model outputs a dog instead of a computer. In a Byzantine

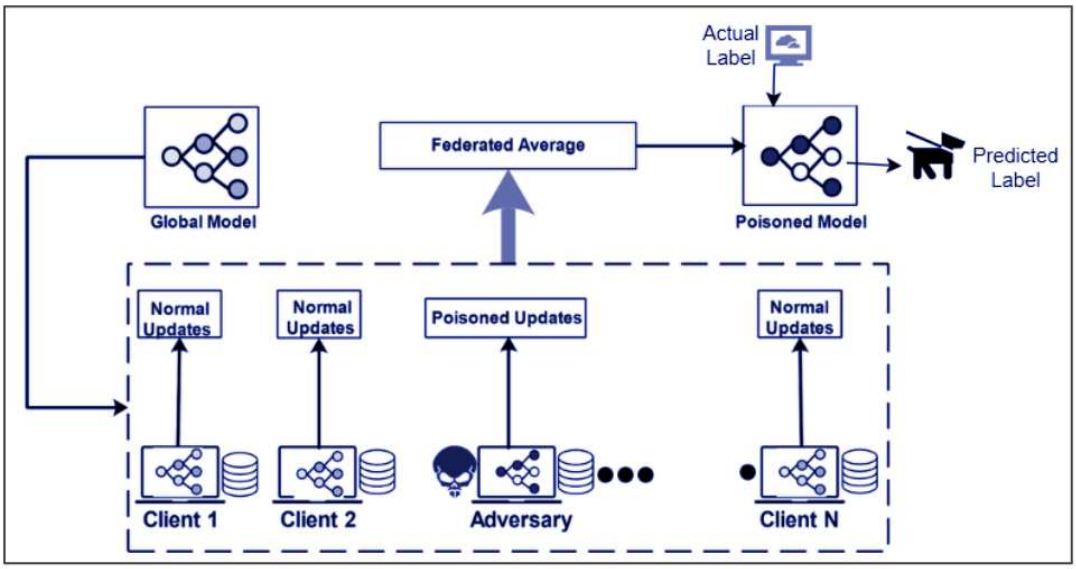


Figure 2.4: Model Poisoning Attack.

attack such as this, some participating clients behave dishonestly or maliciously. These clients may intentionally send incorrect or misleading updates to the central server aggregating the model. This attack aims to undermine the accuracy and integrity of the learned model. Since FL operates in a distributed setting, it is difficult to identify and isolate malicious clients, making it challenging to defend against Byzantine attacks.

In order to mitigate the Byzantine attack, several defence mechanisms have been proposed, such as robust aggregation, SMPC, secure aggregation, client reputation Systems, and so on. The following segments discusses some of these approaches.

2.3.1 Robust Aggregation

FL is a distributed ML mechanism, and each client trains a model in their local data. Thus, to get a concrete global model, the model parameters from the clients need to be aggregated. More robust aggregation techniques can be employed instead of using a simple averaging mechanism to aggregate model updates from clients. These techniques detect and mitigate the influence of

malicious clients by assigning lower weights to their updates or using techniques like Krum [15], FedAvg [79], Trimmed-Mean [143], etc. Some of these techniques are discussed below:

Trimmed-Mean

The Trimmed-Mean is calculated by removing a certain percentage of the extreme values from both ends of the dataset and then taking the average of the remaining values. This method reduces the impact of outliers while still considering the majority of the data. Equation 1 represents the Trimmed-Mean calculation process [143].

$$\overline{X}_{(p)} \leftarrow \frac{1}{n-2pn_p} \sum_{i=p+1}^{n-p} x_{(i)} \tag{1}$$

Here, $\overline{X}_{(p)}$ represents the Trimmed-Mean, n is the total number of observations, n_p is the number of observations to be trimmed from both ends and $x_{(i)}$ is the i^{th} order statistic.

Federated averaging

Federated averaging (FedAvg) is a communication-efficient algorithm for distributed training with many clients. In FedAvg, clients keep their data locally for privacy protection, and a central parameter server is used to communicate between clients [79]. The algorithm alternates between a few local stochastic gradient updates at client nodes and a model-averaged server update. The rationale behind this generalization is that in Federated Stochastic Gradient Descent (FedSGD) [63], if all local nodes start from the same initialization, averaging the gradients is equivalent to averaging the weights. However, in practice, the local nodes may not start from the same initialization, and the gradients may not be aligned with the weights. Therefore, FedAvg allows local nodes to perform more than one batch update on local data and exchanges the updated weights instead of the gradients.

$$w_{t+1} \leftarrow \frac{\sum_{i=1}^{n} n_i w_{t,i}}{\sum_{i=1}^{n} n_i} \tag{2}$$

Equation 2 presents the federated averaging mathematical expression [79], where w is the model weight, n is the number of clients, n_i is the number of samples held by client i, and t is the current

round of training. This equation computes the weighted average of the model weights across all clients, where the weights are proportional to the number of samples each client holds.

Krum

Krum is a robust aggregation method for FL that aims to tolerate Byzantine participants in a distributed setting by selecting fewer models for aggregation, attempting to exclude malicious participants. Each client computes a local model update in Krum and sends it to the server. The server selects a subset of the clients with the smallest Euclidean distance from the updates of the other clients. The server aggregates the selected updates using a weighted average. Krum effectively mitigates Byzantine attacks and outperforms classical aggregation approaches in terms of robustness when the level of corruption is high while being competitive in the regime of low corruption.

$$w_{t+1} \leftarrow \frac{\sum_{i=1}^{n} n_i w_{t,i} - \text{Krum}(w_{t,1}, \ldots, w_{t,n})}{\sum_{i=1}^{n} n_i - 1} \tag{3}$$

Equation 3 presents the Krum aggregation [15], where w is the model weight, n is the number of clients, n_i is the number of samples held by client i, and t is the current round of training. The function Krum computes the weighted median of the model weights across all clients, where the weights are proportional to the number of samples each client holds. The weighted median is computed by first computing the Euclidean distances between each pair of model weights, then selecting the model weight with the smallest sum of distances to all other model weights.

FLRAM

FLRAM is a robust aggregation technique for FL that employs a set of strategies during the client model gradient uploading phase, such as gradient median estimation, distance detection, and clipping transformations, to filter out benign clients that positively influence the convergence of the global model.

Robust aggregation techniques are particularly useful in situations where the data is prone to errors or when outliers are possibly skewing the results. For example, in financial analyses, where extreme values can significantly affect overall performance indicators, robust aggregation methods

can provide a more accurate representation of the underlying trend.

2.4 Client Selection

Regarding data dissemination and hardware setups, clients in FL show notable variability. The local updates from heterogeneous clients may not be effectively utilized by randomly picking clients in each training cycle, which could lead to decreased model accuracy, a slower rate of convergence, compromised fairness, etc. Diverse client selection methods have been developed to address the FL client heterogeneity problem, demonstrating a potential improvement in performance [39]. Before each training session, client selection algorithms select preferred users to take part in based on factors such as the users' data, device performance, and level of trust. Improved model accuracy, increased fairness, stronger robustness, and reduced training overhead can all be achieved with an efficient FL client selection approach. In the following subsections, we are going to discuss some client selection strategies.

2.4.1 Randomized Sampling

Randomized sampling techniques should be used in the client selection process to prevent targeted attacks. This lessens the possibility that Byzantine clients may coordinate their operations on particular rounds or client subsets. The fundamental concept of Randomized Sampling is to choose a subset of clients at random to take part in the training session. Lowering the influence of malicious clients can aid in lessening the effects of Byzantine attacks. Stratified sampling is one method for carrying out Randomized Sampling. This method divides the clients into strata according to reputation scores. One stratum is designated for clients with high reputation scores and another for those with poor scores. When it is necessary to choose a subset of clients, those clients are chosen at random.

2.4.2 Client Reputation Systems

Client Reputation Systems (CRS) are a class of mechanisms that can be used to defend against Byzantine attacks in distributed systems. The basic idea behind CRS is to assign a reputation score

to each client based on their past behaviour. Clients with high reputation scores are trusted more than those with low scores. This can help to mitigate the impact of Byzantine attacks by reducing the influence of malicious clients. One approach to implementing a CRS is to use a weighted voting scheme [147]. In this scheme, each client is assigned a weight based on its reputation score. When a decision needs to be made, the clients' votes are weighted according to their reputation scores. This can help ensure that the most trustworthy clients make decisions. Another approach is to use a threshold-based scheme [107]. In this scheme, clients are classified into trustworthy and untrustworthy. Clients that are classified as trustworthy are allowed to participate in the system, while those that are classified as untrustworthy are excluded. The threshold for classifying clients can be set based on their reputation scores. There are several challenges associated with implementing a CRS. One challenge is to ensure that the reputation scores are accurate. This can be difficult to achieve in practice, as clients may be able to manipulate their reputation scores. Another challenge is ensuring the system is resilient to collusion attacks, where multiple malicious clients work together to subvert the system.

Despite these challenges, CRS can be an effective mechanism for defending against Byzantine attacks. By assigning reputation scores to clients, CRS can help reduce the impact of malicious clients and improve the system's overall security.

2.5 Secure and Private Communication Channels

Secure and Private Communication Channels (SPCC) are a class of mechanisms that can be used to defend against Byzantine attacks in FL. The basic idea behind SPCC is to ensure that the communication channels between the clients and the server are secure and private. This can help to mitigate the impact of Byzantine attacks by reducing the influence of malicious clients. To ensure secure and private exchange SMPC, and HE are used [52].

Secure Multi-Party Computation

SMPC is a cryptographic technique that enables different parties to perform a computation using their private data without revealing it to each other [156]. SMPC is a subfield of cryptography that

aims to create methods for parties to jointly compute a function over their inputs while keeping those inputs private.

The foundation for SMPC started in the late 1970s with the work on mental poker, cryptographic work that simulates game playing/computational tasks over distances without requiring a trusted third party. Traditionally, cryptography was about concealing content, while this new type of computation and protocol is about concealing partial information about data while computing with the data from many sources and correctly producing outputs.

SMPC provides a protocol where no individual can see the other parties' data while distributing the data across multiple parties. It allows data scientists and analysts to compute privately on the distributed data without exposing it. The computation is based on the secret sharing [53] of all the inputs and zero-knowledge proofs for a potentially malicious case, where most honest players in the malicious adversary case assure that bad behaviour is detected and the computation continues with the dishonest person eliminated or their input revealed. SMPC has been used in various applications such as privacy-preserving data mining, secure auctions, and secure voting [156].

Homomorphic Encryption (HE)

HE is a cryptographic technique that enables computations to be performed on encrypted data without decryption [150]. In FL, HE can be used to protect user data privacy during model training and aggregation. By encrypting the model weights and gradients, clients can securely transmit their updates to the server without revealing their private data [74]. Researchers have proposed various secure aggregation protocols for FL, such as Secure Aggregation via Secret Sharing (SAS), Secure FL framework using HE and Verifiable Computing (HEVC), and Privacy-preserving FL based on Multi-key Homomorphic Encryption (MKHE) [75]. These protocols aim to provide strong security guarantees for FL while preserving the accuracy and efficiency of the learning process.

$$E(m_1) \cdot E(m_2) = E(m_1 + m_2) \tag{4}$$

Here, Eq. 4 demonstrates the additive homomorphic property of HE, where E is the encryption function, m_1 and m_2 are plaintext messages, and $+$ is the addition operation. This additive

homomorphic property allows the encrypted sum of two plaintext messages to be computed by multiplying their corresponding ciphertexts.

Differential Privacy

DP provides provable protection against inference attacks in FL by ensuring model updates reveal minimal information about individual data points. A randomized mechanism $\mathcal{M}$ satisfies (ε, δ)-DP if for all neighboring client update sets $\mathcal{W}, \mathcal{W}'$ differing by one client's data:

$$\Pr[\mathcal{M}(\mathcal{W}) \in S] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(\mathcal{W}') \in S] + \delta \tag{5}$$

FL systems implement DP through three core techniques [3]:

1) **Gaussian Noise Injection**: Clients perturb updates with $\mathcal{N}(0, \sigma^2)$, where $\sigma = \frac{\Delta_2}{\varepsilon}\sqrt{2\ln(1.25/\delta)}$.

2) **Update Clipping**: Enforces L_2-sensitivity bounds via $||\mathbf{w}_i||_2 \leq \tau$.

3) **Privacy Amplification**: Achieved through secure aggregation and subsampling [11].

2.6 Adversarial Training

A method for strengthening ML models' resistance to adversarial attacks is called adversarial training. Adversarial training can be applied in FL to strengthen the security of the model updates during aggregation. Through the creation of adversarial instances and their integration into the training procedure, the model can acquire increased resilience against these types of attacks [99].

Researchers have proposed numerous adversarial training techniques, including Federated Adversarial Training (FAT) [160], Federated Adversarial Learning (FAL) [62], and Federated Adversarial Training with Dynamic Weighting (FedDynAT) [99]. By using these techniques, FL models should be more resilient to several types of adversarial assault, including evasion, inference, and poisoning attacks.

2.7 Model Parameter Recovery

In FL, malicious clients can poison the global model by sending malicious model updates to the server. The robust aggregation strategies remove malicious client updates [34, 15, 79] to recover from poisoning attacks. However, there is a chance that the federated server runs out of minimum samples for aggregation. In this scenario, the federated server has to call for repeated training. Therefore, model parameter reconstruction can recover an accurate global model from poisoning attacks with a small computation and communication cost for the clients. The key idea is that the server estimates the clients' model updates instead of asking the clients to compute and communicate them during the recovery process. In particular, the server stores the historical information, including the global models and clients' model updates in each round, when training the poisoned global model before the malicious clients are detected. During the recovery process, the server estimates a client's model update in each round using its stored historical information [21].

2.8 Byzantine Attack defence Framework

Byzantine attacks are a significant threat to the security and privacy of FL systems. To mitigate this threat, researchers have proposed various defence mechanisms, such as Byzantine-resistant aggregation algorithms, client selection strategies, and secure communication protocols [102, 119]. However, these mechanisms may not be sufficient to provide complete protection against Byzantine attacks.

Researchers have proposed Byzantine attack-proof frameworks that provide strong security guarantees for FL systems to address this issue. These frameworks typically involve a combination of multiple defence mechanisms, such as Byzantine-resilient aggregation, client selection, and secure communication, to achieve robustness against Byzantine attacks [102, 130].

However, existing frameworks utilize techniques that suffer from accuracy and efficiency. Therefore, a robust defence framework that can be efficient and accurate is required.

Chapter 3

Literature Review

In this chapter, we compare and contrast relevant related works of this thesis. We subdivided the related works into client selection, protected communication techniques, and aggregation strategies.

3.1 Aggregation Strategies

In FL [79], the federated server averages model parameters accumulated from participating clients. However, the standard averaging mechanism cannot prevent the impact of malicious updates on the model. Therefore, robust Byzantine aggregation and verification-based strategies have been proposed in the literature to mitigate the effects of poisoning attacks.

In Byzantine attacks, adversaries ultimately control authenticated client devices to alter their behaviour, which can affect the model's accuracy. Krum [15] and Fang *et al.'s* [34] approach can offer defence against Byzantine attackers with a robust aggregation strategy. Krum removes poisoned updates by computing the Euclidean distance between model updates. However, a single model parameter can influence Euclidean distance [34]. Therefore, Bulyan first selects local model updates based on the Euclidean distance. Before aggregation, it identifies the model parameters closest to the median and computes their mean to select as one of the model parameters. However, Bulyan can be affected by attacks that are effective on Krum, as it executes Krum many times and computes the pairwise Euclidean distance. Therefore, Fang *et al.'s* scheme [34] removes outliers in model parameters that negatively impact the accuracy and loss of the model. However, to estimate a

particular benign model parameter, Trimmed-Mean [143] orders the model parameters and discards the smallest and largest values from the model parameters while calculating their mean. Another strategy, named FABA [132], offers fast aggregation against Byzantine attacks by iteratively discarding model updates that are distant from the average model updates. Liu *et al.* [67] discard model updates by detecting the difference between benign and malicious updates using the Pearson correlation coefficient.

Some defence mechanisms [13, 117, 111] compute the negative impact of data samples by considering their contribution to the error rate. These types of strategies remove training samples with a more considerable negative impact. However, another defence mechanism *TRIM* [48] tries to minimize the loss function while inferring a training subset from a given model parameter and discards data samples that do not belong to the inferred subset.

While a verification-based strategy, BAFFLE [9] utilizes a feedback-based learning mechanism using a set of validating clients to determine if global communication round updates have been poisoned. However, validating clients may not detect malicious updates, as attackers might use other datasets or techniques to blend poisoned samples. On the other hand, the FAT technique [160] can predict the gradient using another model to detect and mitigate the impact of the poisoned update. However, this technique requires another model and may need to be revised when applied to a new dataset. In [154] and [55], authors utilize Generative Adversarial Networks (GAN) [28] to generate auditing datasets. Using that dataset along with a classifier, they try to predict adversarial updates. To do so they train a classifier to predict malicious samples [42].

However, GAN behaves differently on IID and non-IID datasets [10, 158]. Also, the different distribution of data of FL affects this solution [1]. Another verification strategy proposed by Yu *et al.* [25] considers a digital signature and a trusted environment to detect any alteration in the model. On the other hand, Ronald *et al.* [30] and Jithish *et al.* [51] trains an SVM classifier to distinguish between benign and malicious updates. FLARE [120] can assign a trust score to the local model updates by calculating the penultimate layer representation vector, showing an enormous difference between benign and malicious models. However, their trust computation often requires significant initial information [18].

Recent advancements have brought forth several key techniques to defend against poisoning

attacks in FL, including Stake [31], Shap [56], and Cluster [50]. Stake utilizes blockchain technology for update aggregation, where clients submit their local updates to a blockchain network. The aggregation occurs on the chain, and designated ”voters” then validate these updates. Acceptance by a majority rewards both voters and proposers, while unaccepted updates result in penalties for the involved parties. While robust, this method's reliance on blockchain can introduce significant computational complexity. In contrast, the Shap technique employs SHAP values to identify telltale signs of poisoning attacks within the feature space. However, this approach requires a reference dataset for SHAP computation, which conflicts with the privacy principles of FL. The Cluster technique, on the other hand, uses source and target neurons as distinctive features for detecting Label Flipping attacks, enabling an HDB-SCAN cluster [142] to differentiate between malicious and benign samples. Yet, clustering techniques are generally limited to common clustering problems, and analyzing the feature space often demands prior knowledge about the dataset. Beyond these, methods like FLGuardian [157] have pioneered layer-wise defence mechanisms, using cosine similarity or Euclidean distance and weighted trust scoring to detect anomalous updates by comparing pairwise similarities across neural network layers. However, FLGuardian struggles against adaptive poisoning attacks that cunningly manipulate gradients to evade distance-based detection. Its reliance on static clustering algorithms may also falter against dynamic attacks that gradually shift malicious updates to mimic benign patterns, especially in non-IID settings where natural layer-wise variations exist. Additionally, such methods can face high computational overhead when scaling to complex models, as pairwise comparisons across all layers and clients become prohibitively expensive. In a different vein, AIDFL [24] introduces a novel information-theoretic framework that leverages conditional entropy and mutual information metrics. These metrics are inherently independent of data distributions and detect poisoning attacks by examining the structural relationships between data and model layers. Unlike traditional methods that employ static clustering or aggregation rules, AIDFL implements a multi-level defence protocol that combines K-means clustering [104] with dynamic anomaly detection based on information flow patterns across network layers. While AIDFL's information-theoretic approach effectively handles non-IID data, its reliance on mutual entropy calculations incurs higher computational overhead, and it lacks explicit client authentication for update

verification. Further defence mechanisms include MSGuard [140], which integrates sign statistics, cosine similarity, and spectral anomaly scores within a Mean Shift clustering model to detect Byzantine attacks without requiring prior knowledge of attacker counts. However, its dependence on gradient magnitude filtering might inadvertently discard legitimate updates in non-IID settings, and the computational cost of multi-feature clustering could impede scalability in large-scale FL systems. In contrast, TDF-PAD [7] utilizes the Interquartile Range (IQR) to categorize models as poisoned, benign, or ambiguous, and subsequently applies Z-score analysis to the ambiguous cases. While its adaptive thresholds enhance robustness in non-IID scenarios, computational costs might slow convergence, and dynamic attacks could still evade detection. Lastly, PurifyFL [93] combines Homomorphic Encryption (HE) with poisoning attack detection through cosine direction analysis of updates. While its single-server design improves practicality by supporting additive and multiplicative ciphertext operations, this approach may inadvertently filter benign updates due to directional thresholds. It can impose significant computational burdens on resource-constrained devices.

While these existing defence mechanisms can defend against particular attack types, they present significant limitations that hinder their effectiveness in real-world FL deployments. Table 3.1 provides a comprehensive analysis of these research gaps and demonstrates how the proposed *FedChallenger* and *SignDefence* techniques address these critical limitations. The study reveals that current approaches suffer from fundamental issues, including the lack of attack propagation mitigation, the absence of multi-layer defence strategies, reactive rather than proactive attack prevention, high computational overhead, and limited adaptability to different deployment environments.

As illustrated in Table 3.1, the existing defence mechanisms do not consider mitigating attack propagation in federated aggregations. Additionally, they do not offer multi-layer defence with challenge-response and aggregation to prevent and mitigate poisoning attacks. The proposed *FedChallenger* addresses these limitations through a zero-trust challenge-response architecture that incorporates continuous authentication and MAD-based robust aggregation. *FedChallenger* utilizes a trust-based attack detection algorithm that relies on challenge-response information to compute trust scores and introduces a robust aggregation mechanism that applies cosine similarity-based consensus boosting to benign weights while dynamically pruning malicious updates.

For environments where challenge-response information accumulation may not be feasible due

Table 3.1: Research Gap Analysis: Comparison of Existing Limitations and Proposed Solutions

Research Gap	Description	Existing Limitations	FedChallenger Solution	SignDefence Solution
Attack Propagation Mitigation	Existing mechanisms don't consider mitigating attack propagation in federated aggregations	Current defences focus on individual attack detection but don't prevent the spread of malicious updates across the federation	Zero-trust challenge-response architecture that actively authenticates all participating devices before and during training sessions	Sign-magnitude gradient analysis prevents propagation by identifying malicious patterns at the gradient level
Multi-layer defence	Lack of comprehensive multi-layer defence combining challenge-response and aggregation mechanisms	Most solutions provide single-layer protection (either detection OR aggregation, not both)	Dual-layer approach: Challenge-response authentication + robust aggregation with MAD-based trimming	Lightweight multi-layer: Sign-magnitude analysis + LeakyReLU with Jaccard Similarity for weight estimation
Real-time Attack Prevention	Existing methods are reactive rather than proactive in preventing attacks	Solutions like Krum [15], Bulyan [81], and Trimmed-Mean [143] detect attacks after they occur	Proactive prevention through continuous device authentication and trust computation	Immediate detection through real-time gradient sign analysis
Computational Efficiency	Heavy computational overhead in existing robust aggregation methods	Methods like Bulyan require multiple Krum executions and pairwise distance calculations	Cosine similarity-based consensus boosting reduces computational complexity while maintaining robustness	Lightweight design using sign-magnitude operations, significantly reducing computational burden
Dynamic Threshold Adaptation	Fixed threshold-based approaches cannot adapt to varying attack intensities	Solutions like FABA [132] and correlation-based methods use static thresholds	MAD-based dynamic pruning adapts thresholds based on statistical deviation of updates	Adaptive sign-magnitude thresholds adjust based on gradient patterns
Trust-based Authentication	Lack of continuous trust assessment for participating devices	Existing solutions assume device authenticity after initial verification	Trust-based attack detection algorithm continuously computes and updates device trust scores	Implicit trust through gradient consistency - devices with consistent sign patterns are deemed trustworthy
Communication Overhead	Heavy communication requirements for validation and verification	Solutions like BAFFLE [9] require additional validating clients, increasing communication burden	Optimized challenge-response protocol minimizes communication overhead	Minimal communication - uses only sign information of gradients, reducing data transmission
Deployment Flexibility	Limited adaptability to different deployment environments	Most solutions require specific infrastructure or additional models (like FAT [160] requiring another model)	Configurable architecture with two versions (fixed and dynamic threshold) for different deployment scenarios	Environment-agnostic design that works across various deployment contexts without additional infrastructure
Attack Type Generalization	defence mechanisms are specialized for particular attack types	Solutions are effective against specific attacks but vulnerable to others	Comprehensive defence against multiple attack types through combined authentication and aggregation	Broad spectrum protection effective against Min-Max, Min-Sum, and Label Flipping attacks
Performance Under non-IID Data	Limited effectiveness in FL's inherent non-IID data distribution	GAN-based solutions [154, 55] and other methods perform poorly with non-IID data distributions	Robust performance across different data distributions through statistical aggregation methods	Distribution-agnostic approach using sign-magnitude analysis that works regardless of data distribution

to deployment constraints or communication burden, *SignDefence* provides a lightweight alternative. *SignDefence* leverages sign-magnitude of gradients with LeakyReLU and Jaccard Similarity

for weight estimation to effectively mitigate poisoning attacks while maintaining minimal communication overhead. This approach offers broad spectrum protection against various attack types including Min-Max, Min-Sum, and Label Flipping attacks, making it suitable for diverse deployment scenarios.

Both proposed solutions represent significant advances over existing approaches by providing comprehensive, efficient, and adaptable defence mechanisms that address the fundamental research gaps identified in current FL security literature, as comprehensively analyzed in Table 3.1.

3.2 Client Selection Approaches

Client selection is a critical stage of FL, and improper client selection can lead to the inclusion of malicious clients. This section presents a detailed examination of various client selection algorithms, highlighting their strengths, weaknesses, and the implications of their application in FL environments. Client selection algorithms are subdivided into randomized selection [91], weighted sampling [159], performance-based selection [39], reinforcement learning-based selection [95], and reputation-based selection [77]. A comprehensive comparison of these approaches and their limitations is presented in Table 3.2, while the specific research gaps addressed by *Fed-Reputed* are detailed in Table 3.3.

Randomized client selection algorithms randomly choose a subset of clients for model updates in each round. These algorithms are simple to implement and require minimal communication overhead [91]. Randomization can also provide fairness in client participation [23]. However, the lack of control over the quality of client updates and potential imbalances in the distribution of updates may result in slow convergence and suboptimal model performance [124, 39]. Additionally, Lei et al. [39] suggest that clients cannot be randomly selected; instead, they should be chosen based on their utility.

Weighted client selection algorithms [159, 38] assign different weights to clients based on their characteristics, such as available computational resources or data distribution. They enable the prioritization of more capable or representative clients, potentially improving overall learning efficiency and model performance [159]. However, determining appropriate weights and accurately

representing client capabilities can be challenging [72]. Additionally, weighted sampling may lead to biased model updates if not carefully calibrated [38].

On the other hand, performance-based algorithms select clients based on their historical performance, considering factors such as model update quality and convergence speed [39]. Using the client's performance history can help identify high-quality contributors, thereby enhancing the overall learning process [94]. However, the need for continuous monitoring and evaluation of client performance may introduce additional computational overhead [4]. Moreover, clients with poor initial performance might be excluded, potentially limiting the exploration of the global model space [103].

Some proximal algorithms consider both local and global model parameters, penalizing clients whose updates deviate significantly from the global models. They can promote model consistency between clients, reducing the risk of model divergence [17]. However, determining appropriate penalty functions and tuning hyperparameters can be non-trivial [145]. Additionally, these methods may be sensitive to outliers, which can affect convergence.

Reinforcement learning-based [149] client selection algorithms employ reinforcement learning techniques to adapt the client selection strategy dynamically. Adaptive client selection allows for real-time adjustments based on the evolving FL environment. A DRL approach proposed in [95] considers heavy resource consumption as a sign of attacker presence. They even consider longer training times to be suspicious behaviour. Their DRL algorithm utilizes Q-Learning and identifies trusted and non-trusted clients. However, the approach did not address the solution to the cold-start problem. Training effective reinforcement learning models for client selection requires significant computational resources [151]. Moreover, the interpretability and transparency of such models may be limited [12].

Another technique called FedMCCS [5] models the FL client selection as a multi-objective regression function and filters out clients based on training time, memory, CPU, and energy consumption. Another energy-saving approach, called FedLE [128], considers the available battery life of edge-IoT devices and utilizes a Convolutional Neural Network (CNN) to select the proper client. They train part of the model to identify similar clients and cluster them, ensuring that similar clients are preserved for backup selection and preventing the selection of low-powered clients.

Another cluster-based approach named Flame [86], group similar updates using HDBSCAN with cosine distance before applying adaptive noise and clipping. While Flame enhances resilience against certain attacks, its reliance on cosine similarity in high-dimensional spaces can result in misclustering under sign-flipping attacks, and its noise injection may degrade model performance.

A Multi-Armed Bandit approach [8] considers training time and local loss as criteria for client selection, and based on these criteria, it rewards clients. FedMiNT [125], on the other hand, employs a matching strategy for client association with the proper server and utilizes a bootstrapping server to gather initial training accuracy. However, these approaches could not resolve the cold-start problem [125] correctly, and they are specific to particular client selection problem types. Therefore, *FedReputed* has been proposed as a standalone solution for client selection, considering the cold-start problem and the detection of clients' misbehaviour.

Client reputation systems can be a practical approach to address the challenges of client selection in FL [124]. They may help identify honest and dishonest clients within the network, which can enhance security mechanisms and learning performance. Client reputation is often determined by the performance and contribution of the client to the environment. These measurements can be used to identify honest (high contribution) and dishonest (low contribution) clients in the network, which can help security mechanisms to be more effective [122]. In a different approach [134], a reverse auction mechanism was implemented alongside an incentive system for reputable clients. This work used a Deep Reinforcement Learning (DRL) algorithm to select highly reputable clients optimally.

A reputation-aware stochastic integer programming-based FL client selection method (SCS) [113] considered client selection cost and modelled the client selection problem as a stochastic integer program (SIP) to handle the uncertainties in proper client selection, including the cold-start problem. It also considers optimal client selection from both low- and high-reputation pools to ensure fairness in selection. However, this approach employed a third-party defence called FoolsGold [40] for attack detection and prevention. Moreover, SIPs lead to additional complexity in the FL training process and incur high computational costs. Alternatively, the MADDPG approach [77] utilizes the DRL mechanism known as Multi-Agent Deep Deterministic Policy Gradient to adjust the reputation threshold. However, this approach did not effectively address the cold-start problem. Additionally, it relied on a third-party defence mechanism for malicious node detection.

Table 3.2: Comparison of Client Selection Techniques and Research Gaps Addressed by Fed-Reputed

Technique	**Key Approach**	**Strengths**	**Limitations**	**Gap Addressed by Fed-Reputed**	**Cold-Start Handling**
Randomized Selection [91]	Random subset selection	Simple implementation, fairness	No quality control, slow convergence	Device capability-aware selection	No
Weighted Sampling [159]	Weight-based prioritization	Resource-aware selection	Difficult weight calibration, bias risk	Automated weight learning via DQN	Partial
Performance-based [39]	Historical performance metrics	Quality contributor identification	Excludes poor initial performers	Balanced exploration-exploitation	No
Proximal Algorithms [17]	Local-global parameter deviation	Model consistency promotion	Hyperparameter sensitivity	Adaptive penalty via Bellman equation	No
DRL-based [95]	Q-Learning for trust assessment	Adaptive real-time selection	High computational cost, cold-start issue	ICMDP with hierarchical framework	No
FedMCCS [5]	Multi-objective regression	Resource consumption filtering	Specific to resource constraints	Comprehensive misbehavior detection	No
FedLE [128]	CNN-based clustering	Energy-aware selection	Limited to battery-powered devices	Universal device capability handling	Partial
Multi-Armed Bandit [8]	Reward-based selection	Training time optimization	Limited exploration strategy	Enhanced exploration via ICMDP	No
FedMiNT [125]	Matching strategy	Server-client association	Bootstrapping dependency	Self-contained reputation system	Partial
SCS [113]	Stochastic integer programming	Fairness guarantee	High complexity, third-party dependency	Independent defence mechanism	Yes
MADDPG [77]	Multi-agent DRL	Adaptive threshold	Third-party dependency, cold-start	Integrated malicious detection	No
Fed-Reputed	**DQN-based ICMDP with modified Bellman equation**	**Comprehensive misbehavior detection, cold-start handling, device-aware selection**	**-**	**All above gaps addressed**	**Yes**

Table 3.3: Key Research Gaps and Fed-Reputed Solutions

Research Gap	**Existing Limitations**	**Fed-Reputed Solution**
Cold-Start Problem	Most techniques fail to handle new clients without historical data	Hierarchical framework with device capability bootstrapping
Malicious vs Straggler Differentiation	Existing methods treat both as similar problems	ICMDP classifier specifically designed for misbehavior classification
Third-Party Dependency	Many approaches rely on external defence mechanisms	Self-contained reputation and detection system
Herding Problem	High-reputation clients are over-selected	Modified Bellman equation balances exploration-exploitation
Device Heterogeneity	Limited consideration of diverse device capabilities	Comprehensive device capability information integration
Computational Complexity	Complex mathematical models (SIP, MADDPG)	Efficient DQN-based approach with manageable complexity
Fairness vs Performance Trade-off	Difficulty in balancing fair selection with optimal performance	Reputation-based fair selection with performance guarantees
Real-time Adaptability	Static approaches cannot adapt to dynamic FL environments	Dynamic reputation updating with Markov Decision Process

As illustrated in Table 3.2, existing client selection techniques suffer from various limitations, including the inability to handle cold-start scenarios, difficulty in differentiating between malicious and straggler clients, dependency on third-party mechanisms, and computational complexity issues. Table 3.3 specifically outlines how *Fed-Reputed* addresses these fundamental challenges through its comprehensive approach, incorporating device capability information, modified Bellman equations, and ICMDP-based classification. The proposed *Fed-Reputed* technique addresses multiple research gaps simultaneously, offering a comprehensive solution that existing approaches handle only partially or not at all.

3.3 Inference Attack-Resistant Techniques

Secure Multi-Party Computation (SMPC) serves as a fundamental component in safeguarding privacy and security within FL, a collaborative machine learning paradigm that encompasses multiple participating entities. SMPC-integrated FL frameworks typically require several rounds of

communication between participants to achieve secure parameter aggregation, which remains fundamental for mitigating inference attacks [74]. Vertical Federated Learning (VFL) has emerged as a practical approach to multi-party secure computation in machine learning contexts, facilitating the acquisition of comprehensive datasets while maintaining the privacy of individual data custodians [59]. Furthermore, FL implementations frequently require the incorporation of multi-party computation methodologies to deliver robust privacy assurances, mitigating potential inference attacks and ensuring that untrusted aggregation servers cannot access isolated responses from participating entities during the training phase [136].

One notable approach [109] enables the execution of distributed machine learning in a privacy-preserving framework, where end-hosts remain unaware of clients' actual model architectures. An alternative methodology involves utilizing blockchain technology to enhance FL security [138]. Blockchain-integrated FL systems can protect individual data point privacy while ensuring that model parameters do not disclose information regarding the specific data points employed in local model training [90]. The BlockchainMPC framework [52] introduces a decentralized, verifiable FL architecture that leverages blockchain technology for secure aggregation and tamper-resistant storage, while critically employing SMPC for privacy-preserving model verification, directly eliminating poisoned models and protecting local parameters from inference and stealing attacks.

This methodology prioritizes privacy preservation while eliminating compromised local models through an encrypted inference process via SMPC for verification prior to aggregation, protecting membership inference and parameter-stealing attacks. However, this technique demands substantial computational resources and requires extended convergence times, as highlighted in Table 3.4.

Another significant technique for SMPC implementation in FL involves HE. HE represents an encryption paradigm that enables computations to be executed on encrypted data without requiring prior decryption. This capability proves particularly valuable in FL contexts as it permits model parameters to be encrypted before transmission to central servers, thereby ensuring the protection of individual data point privacy. López-Al et al. [71] proposed an NTRU encryption system [44] based on LTV12, which utilizes linearization and modulus switching technologies to achieve a levelled multi-key fully HE scheme. However, this approach suffers from the substantial computational complexity associated with decryption operations. While more efficient protocols, such as

DHS16 [33], have been proposed, NTRU-based MK-FHE schemes remain impractical due to their considerable decryption complexity and high communication overhead.

The application of alternative types of FHE schemes for MK-FHE was initially discussed by Michael et al. [71]. Further simplifications in the ciphertext extension process and multiple multi-party computations (MPC) rounds have been introduced [36]. Nevertheless, the ciphertext size in these schemes increases exponentially with the number of participants, resulting in infeasible communication and storage costs for practical implementation scenarios. In 2019, Chen et al. [69] proposed MK-BFV and MK-CKKS, multi-key variants of the fully HE schemes BFV [127] and CKKS [110]. In these schemes, the ciphertext length increases linearly with the number of participants. The implementations of MK-BFV and MK-CKKS provide the first experimental results of MK-HE with packed ciphertexts. However, the distributed decryption of a multi-key ciphertext requires an additional secure method, as it poses the risk of privacy leakage.

Beyond the realm of FHE, other privacy-preserving techniques in FL face their own set of challenges. For instance, secure differentially private stochastic gradient descent (DPSGD), as demonstrated by Ruan et al. [96], can effectively prevent membership inference attacks. The authors claim their approach minimizes accuracy loss caused by differential privacy (DP) through local data-based model initialization and data-independent feature extraction.

However, recent studies have revealed unexpected vulnerabilities even in these privacy-preserving FL methods. While DP is designed to protect against inference attacks, Hossain et al. [45] show that it can inadvertently enable stealthy model poisoning (DeSMP). The DP noise can mask malicious updates, making them harder to detect. To counter this, they propose a reinforcement learning-based defence that dynamically adjusts DP noise levels, albeit introducing latency due to iterative policy updates. Complementing these efforts, Zhi et al. [35] presented a dual-party secure communication framework utilizing a semi-homomorphic Gradient Boosting Machine (SecureGBM). While they claimed to reduce communication overhead caused by Homomorphic Encryption (HE) through stochastic approximation, their approach still incurred a 3% accuracy loss. These contrasting findings highlight the complex trade-offs between privacy, security, and performance in designing robust FL systems.

Another gradient boosting strategy, SimFL [61], utilizes locality-sensitive hashing to determine

data similarity. The experimental evaluation suggests accuracy improvement, but the approach is limited to horizontal FL applications.

In another investigation, Shi et al. [101] proposed an efficient Multi-participant Vertical FL based on Secret Sharing (MVFLS). Secret sharing replaces HE in their improved multi-party VFL, significantly enhancing communication and computation efficiency. In a separate study, Wang et al. [121] proposed a novel FL scheme based on SMPC and DP. The scheme prevents inference during both the learning process and output inference phases. Maurya et al. [78] developed an SMPC-based healthcare framework with dynamic edge thresholding, though communication overhead and data heterogeneity issues persist.

Another approach to SMPC implementation in FL involves secure aggregation [106]. Secure aggregation is a technique that enables central servers to aggregate model parameters from all parties without revealing individual model parameters. A recent study proposed a secure aggregation framework, Multi-RoundSecAgg [106], with multi-round privacy guarantees. This framework introduces a novel metric to quantify the privacy guarantees of FL over multiple training rounds. It develops a structured user selection strategy that guarantees the long-term privacy of each user across any number of training rounds. The framework also carefully considers fairness and the average number of participating users at each round. Another paper [16] proposes a practical and secure aggregation protocol for FL on user-held data. It considers training a deep neural network in the FL model using distributed stochastic gradient descent across user devices. The proposed protocol allows a collection of mutually distrustful parties, each with a private value, to collaboratively compute the sum of those values without revealing the values themselves.

However, despite the potential of SMPC in FL, several limitations and challenges require attention. Existing studies have identified that current SMPC approaches need to be more efficient and effective for practical implementation. Additionally, for certain types of computation, the final result can reveal information about the inputs, posing a significant challenge to the privacy guarantees provided by SMPC. Furthermore, the efficiency and scalability of SMPC-based FL remain areas of concern, particularly in industrial-scale applications, as detailed in Table 3.4.

To address these limitations, researchers have proposed various approaches and frameworks.

Table 3.4: Research Gaps in Existing Techniques and Solutions Provided by Ada-Sign and SignMPC

Technique	Key Research Gaps	How Ada-Sign Addresses	How SignMPC Addresses
SMPC-FL [74]	High computational overhead, slow convergence, lacks adaptive mechanisms	DP integration provides privacy with lower overhead, adaptive threshold eliminates manual tuning	Combines HE+DP+SMPC for comprehensive protection, adaptive aggregation improves convergence
Blockchain MPC [52]	Extremely resource-intensive, impractical for real-world deployment	Lightweight sign-based operations with DP protection	Eliminates blockchain overhead while maintaining security through multi-layered HE+DP+SMPC approach
HE [71]	Exponential complexity growth, ciphertext size explosion, poor scalability	Avoids HE entirely, uses DP for privacy with sign-based efficiency	Optimized HE integration within multi-layered framework, balanced with DP and SMPC
MK-BFV/CKKS [69]	Complex decryption requirements, privacy leakage risks, high overhead	Eliminates complex encryption, DP provides sufficient privacy guarantees	Strategic HE use combined with DP and SMPC eliminates decryption vulnerabilities
SecureGMBM [35]	Accuracy degradation (3% loss), still dependent on HE overhead	Maintains accuracy through leaky weighted aggregation, no HE dependency	Comprehensive protection maintains accuracy while providing stronger security than HE-only approaches
MVFLS [101]	Limited to vertical FL scenarios, lacks horizontal FL support	Adaptive mechanisms work for both horizontal and vertical FL settings	Multi-layered approach supports all FL architectures with comprehensive protection
Multi-RoundSecAgg [106]	Complex user selection strategy, requires careful coordination	Eliminates need for complex selection through adaptive threshold mechanisms	Continuous protection without user selection complexity through unified framework
DeTrust-FL [137]	Decentralized coordination complexity, scalability challenges	Centralized efficiency with distributed privacy through DP	Maintains centralized efficiency while providing distributed security guarantees
FLAD [114]	Requires server dataset for training, limited to detected attack patterns	No server dataset requirement, adaptive detection through Jaccard similarity	Comprehensive protection against all attack types, not limited to pre-trained patterns
DPSGD [96]	Accuracy degradation from DP noise, single-layer protection	Efficient DP integration with sign-based aggregation minimizes accuracy loss	Multi-layered protection (HE+DP+SMPC) provides stronger guarantees than DP alone

For instance, DeTrust-FL [137] has been introduced as an efficient, scalable, and secure aggregation-based privacy-preserving FL decentralized trust approach, aiming to resolve the dilemma of secure multi-party aggregation efficiency and centralized trust. Another trust-based aggregation called FLTrust [22] evaluates local updates by comparing them to a root model using cosine similarity and ReLU-based clipping. While effective in some scenarios, FLTrust's static trust threshold struggles with non-IID data distributions. Additionally, a secure federated transfer learning framework has been developed [68], requiring minimal modifications to the existing model structure and providing the same level of accuracy as the non-privacy-preserving approach. Furthermore, efforts have been made to extend SMPC schemes to achieve improved efficiency and privacy guarantees.

Another technique named FLAD [114] offers Byzantine-robust FL against model poisoning by detecting malicious gradients. It uses neural networks (FEMs) trained on a small server dataset to extract gradient features. Malicious gradients are identified by comparing their features to server dataset criteria using clustering (e.g., DBSCAN) and similarity metrics (such as cosine similarity and relative length difference). This enables FLAD to filter out poisoned contributions, ensuring robust model aggregation even in the presence of numerous attackers. PFLAD further enhances privacy by employing CKKS homomorphism [26] and Random Permutation to protect transmitted gradients.

A sign-based approach named Flod [32] converts model updates into binary representations and computes a weighted average based on Hamming distance, discarding outliers via τ-clipping. Despite its efficiency, Flod's dependence on Hamming distance makes it sensitive to sparse gradients, and its ReLU-based aggregation suffers from the dying ReLU problem, leading to irreversible neuron deactivation.

Another well-known defence strategy called FoolsGold [40] is proposed to defend Sybil-based poisoning attacks in FL, which adapts client learning rates based on contribution similarity to identify and mitigate malicious behaviour. Unlike prior defences that require explicit bounds on the number of Sybils, FoolsGold [40] operates without such assumptions and is shown to be effective against Label Flipping and backdoor attacks across diverse datasets and model types. However, it is essential to note that FoolsGold [40] is not designed to mitigate attacks from a single malicious client, where techniques like Multi-Krum might be more effective.

In conclusion, the literature highlights the significance of SMPC in FL, emphasizing its role in preserving privacy, preventing inference attacks, and ensuring the security of collaborative machine learning. The integration of HE and DP into FL frameworks demonstrates the ongoing efforts to address privacy and security concerns in multi-party FL collaborations. The research gaps identified in Table 3.4 indicate areas where *Ada-Sign* and *SignMPC* play a crucial part.

Chapter 4

FedChallenger: A Robust Challenge-Response and Aggregation Strategy to Defend Poisoning Attacks in Federated Learning

This chapter introduces *FedChallenger*, a dual-layer defence framework that detects and prevents malicious client participation in FL training. The first layer employs zero-trust challenge-response authentication, while the second layer utilizes an enhanced Trimmed-Mean aggregation strategy incorporating pairwise cosine similarity and Median Absolute Deviation (MAD) to mitigate malicious model parameters. The following subsequent sections details the motivations and functional components of the proposed *FedChallenger* approach.

4.1 Motivations and Contributions

As discussed earlier, FL ecosystems are increasingly targeted by evolving model poisoning and data poisoning attacks [131], specifically threatening consumer privacy [112]. In model poisoning, adversaries manipulate model parameters to degrade overall performance. Conversely, data

poisoning involves adversaries altering training data labels, causing the model to learn incorrect predictions for specific samples. These manipulations typically rely on data sampled from Gaussian distributions or computed via inverted loss functions [43]. Both attack types can lead to significant accuracy degradation, potentially resulting in severe consequences and undermining the fundamental benefits of FL-based systems, rendering the technology obsolete. The proliferation of attack incidents targeting FL-based system privacy has led to the proposal of numerous central server-based aggregation mechanisms designed to mitigate their impact. Notable contributions in this area include Krum [15], FedAvg [79], Trimmed-Mean [143], and Fang *et al.*'s [34] defence strategies, with the latter combining ERR and LFR approaches, subsequently referred to as DUEL.

Current defences against model and data poisoning attacks employ diverse aggregation strategies, each with inherent limitations. FedAvg, the foundational approach, naively averages all client updates without discrimination, making it susceptible to basic poisoning where malicious clients submit manipulated gradients. Krum enhances resilience by selecting updates closest to their neighbours through Euclidean distance minimization; however, it struggles against coordinated poisoning attacks where multiple adversaries craft seemingly legitimate updates that collectively distort the global model. Trimmed-Mean addresses outliers by eliminating extreme values based on median deviation, but its fixed trimming ratio cannot adapt to dynamic poisoning strategies that gradually introduce bias. The DUEL approach offers an innovative methodology for detecting malicious samples through parameter-wise loss impact analysis. Yet, like other existing defence mechanisms, it employs a reactive strategy, only mitigating poisoning attacks after they have already compromised the training process. This fundamental limitation persists because current methods lack proactive safeguards to prevent poisoned data from initially contaminating the training pipeline. Furthermore, DUEL's effectiveness is constrained by two critical factors: (1) the loss function's capacity to represent model performance accurately and (2) the tendency of error-based rejection mechanisms to discard legitimate predictions alongside malicious ones inadvertently.

Most cutting-edge approaches defend against poisoning attacks through either robust aggregation [15, 143, 34] or verification-based rejection [154, 9]. Robust aggregation strategies depend on outlier removal using statistical techniques. In contrast, verification-based strategies identify

changes in model updates and data integrity issues using ML or cryptographic techniques to eliminate attack samples. However, these approaches still lack precision and cannot entirely defend against poisoning attacks. Moreover, they seldom provide any mechanism to *prevent* these attacks, allowing attackers' modified samples to propagate into the aggregation computation.

Recent hybrid defence techniques such as Stake [31], Shapley-based methods (hereafter Shap) [56], and Cluster-based approaches (hereafter Cluster) [50] each employ distinct strategies against poisoning attacks but encounter two significant challenges: high computational costs and reduced accuracy with non-IID data distributions. The Stake method employs blockchain technology for majority voting, combined with client reward and penalty calculations, which improves security but introduces substantial processing delays. Shap depends on SHAP value computations, creating considerable computational demands, particularly for complex models. Meanwhile, the Cluster technique analyzes gradients from client updates to identify source and target classes before applying HDBSCAN [142] clustering to detect malicious updates. This approach demonstrates particular sensitivity to variations in data distribution. While these methods advance poisoning detection, their practical deployment remains limited by these computational and data distribution constraints in real-world FL scenarios. Since state-of-the-art techniques do not guarantee a poisoning-free environment, a robust defence mechanism capable of detecting, preventing, and efficiently recovering from such attacks remains essential.

This chapter proposes an extension of our previously designed *FedChallenger* approach [82], which can detect and prevent malicious participants from engaging in FL training using challenge-response mechanisms. The extended approach introduces a revised zero-trust challenge-response architecture [88] that actively authenticates all participating devices before and during training sessions. The framework incorporates an enhanced robust aggregation algorithm utilizing Median Absolute Deviation (MAD)-based [46] trimming to strengthen resilience against poisoning attacks. Furthermore, the performance of the extended version is evaluated across multiple benchmark datasets [135], including comparisons with the most recent techniques [31, 56, 50].

The key contributions of this research are as follows:

- Propose a zero-trust challenge-response-based defence mechanism named *FedChallenger* to detect and prevent poisoning attacks.

- Present a trust-based attack detection algorithm that relies on challenge-response information to compute trust.
- Introduce a robust aggregation mechanism that applies cosine similarity-based consensus boosting to benign weights and dynamically prunes malicious updates via MAD for adversarial updates.
- Evaluate the performance of the proposed approach on MNIST, FMNIST, EMNIST, and CIFAR-10 datasets [135] using different evaluation metrics such as convergence time and accuracy.

4.2 Proposed FedChallenger Design and Architecture

This section outlines the architectural and design aspects of the proposed *FedChallenger* technique, which aims to mitigate poisoning attacks.

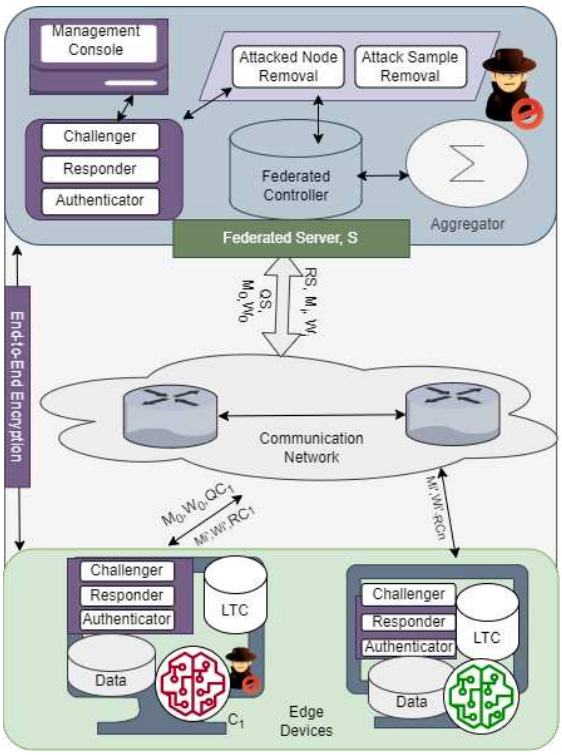


Figure 4.1: The Architecture of FedChallenger.

FedChallenger adopts a modular zero-trust [88] architecture, comprising core components: the Challenger, Responder, Authenticator, and Aggregator. Fig. 4.1 illustrates the overall architecture of the proposed *FedChallenger* strategy.

The architecture arranges the Federated server, S, Communication Network, CN, and Edge Devices, ED, in a hierarchical structure. Within this configuration, both ED and S incorporate dedicated challenge-response modules. These modules are composed of Challenger, Responder, and Authenticator submodules. The Challenger submodule poses questions to the target device, while the Responder submodule provides answers to inquiries from the Challenger submodule of another device. The Authenticator submodule is responsible for making decisions and verifying devices based on their challenge-response interactions.

The management console at S defines the criteria for the questions to be posed. The federated controller, FC, orchestrates model training, removes malicious nodes, eliminates attack samples, aggregates model updates, and implements management decisions. The process of eliminating attacker nodes and attack samples is coordinated among the FC, the challenge-response module, and their respective submodules. Each ED is equipped with a Local Training Controller, LTC, and a neural engine. These components utilize initial model parameters and receive training instructions from the FC. Using this preliminary information, their LTC manages model training with their respective local datasets. The refined model parameters are then transmitted back to the S for aggregation. The FC aggregates model parameters once received from multiple participants through the aggregator submodule. Should the FC identify any malicious model updates or devices, they are immediately isolated upon detection to prevent their adverse impact on the training process. The aggregator submodule can also filter out anomalies originating from adversarial participants.

At the commencement of the training process, the management console instructs the FC to disseminate the details of the model training and to gather participants from the EDs. Interested participants, denoted as $I = (i_1, i_2, ..., i_n) \in ED$, respond to the FC's request and establish a shared secret key, $K_{i,S}$, for their subsequent communications.

Following this, each interested participant, i, and S independently generate a set of management-defined questions, $Q_{i,S}$ and $Q_{S,i}$ respectively, to construct their corresponding challenge matrices, $CM_{i,S}$ and $CM_{S,i}$, for future challenges. These questions cover a spectrum of system attributes,

including CPU speed, memory capacity, fan speed, data folder size, and cryptographic hash values of data points at randomly chosen indices. All participants possess their peer's challenge matrix, $CM_{i,S}$ and $CM_{S,i}$, which contain unaltered challenge-response data. This information is subsequently utilized to detect changes in a participant's behaviour, potentially leading to the early identification of malicious entities. With the questions and answers in place, the FC initiates the training rounds.

In the initial round, the FC challenges participants with random questions $Q_{S,i'}$ drawn from its question pool. Each participant responds with $R_{i',S}$ and, in turn, poses questions $Q_{i',S}$ to the S. The FC cross-references the received answers with its pre-established challenge matrix CM_S. In the event of a discrepancy, the FC removes the corresponding participants from its list and blocks further interactions, coordinating with the attacker node removal submodule. For the remaining participants, the FC provides the answers $R_{S,i'}$ to each of their questions and transmits the initial model M_0 and weights W_0. Upon receiving this information, each participant I first verifies the server's Response $R_{S,i'}$ against their stored $CM_{i',S}$. If a mismatch is detected, they may opt out of the training process.

Algorithm 1 Malicious Device Detection

Input:
Decrypted Challenge, $C'_{i,S}$
Shared secret ($K_{i,S}$), Challenge Matrix (CM), Expected Results ($ER_{i,S}$), Similarity Threshold (ST), Trust Factor (TF) Trust Score ($T_{i,S}$) & Trust Threshold (TTH)
Output:
Malicious Status (M_i)

Selected Challenge, $CS \in Rand(CM, Total(CM))$
for each $c \in CS$ **do**
  Encrypted Challenge, $EC_{i,S} = K_{i,S}(c)$
  Response, $R = challengedEntity(EC_{i,S})$
  Expected Response, $C'_{i,S} = ER_{i,S}(c)$
  Response Similarity, $C_{sim} = similarity(C', R)$
  if $C_{sim} > ST$ **then**
    $T_{i,S} = T_{i,S} \times TF$
  else
    $T_{i,S} = T_{i,S}/TF$
  end if
end for

if $T_{i,S} > TTH$ **then**
  $M_i = False$
else
  $M_i = True$
end if
Return M_i

Otherwise, they proceed to use the initial model M_0 and weights W_0 to commence training. The LTC at each I utilizes these initial parameters and their local data to train the model using Stochastic Gradient Descent (SGD) [70].

4.2.1 Malicious Device Detection:

Algorithm 1 outlines the strategy for detecting malicious devices across all client and server components. For identifying malicious clients, the S challenges participant I with a selected challenge set $CS \in CM$ to receive its Response, R. Each challenge $c \in CS$ is encrypted using a shared secret $K_{i,S}$ between the client and server prior to transmission, ensuring the confidentiality and integrity of the challenge.

Subsequently, the S calculates the similarity between R and the Expected Response, $C'_{i,S}$, for the corresponding challenge, $c \in CS$, where the $ER_{i,s}$ module accumulates the responses. A higher similarity, when compared to a management-defined threshold, ST, leads to a multiplicative increase in trust by a trust factor TF; conversely, trust is penalized by the same factor. After computing the overall trust for each CS, the trust threshold determines the malicious status of the device.

4.2.2 Client-side Defence:

Algorithm 2 illustrates the client-side defence and training procedures.

This algorithm executes challenge-response modules to enable a challenge-response-based defence through a question-answering approach. Initially, client i expresses its interest in participating in the training to the S via a training request TRR_i. Successful acceptance of this request leads to the establishment of shared secrets, $K_{i,S}$, and initial model-related information. During the training process, the detection of malicious activity using Algorithm 1 results in a reduction of the S's trust score, leading to a halt in training. Otherwise, the mini-batch SGD determines the model weights W' for each batch, $b \in B$. Finally, these weights are returned to the S for subsequent processing.

Algorithm 2 Training with Defence at Client, i

Input:

Batches, B

Model, M'

Weights, W'

Output:

Weight (W'_B)

Training Request, $TRR_i = RequestToParticipate(S)$
Run Parallel $ChallengeResponseModule()$
Shared Secret, $K_{i,S} = establishSharedSecret()$
if TRR_i **then**
  if $isMAlicious(S, K_{i,S})$ **then**
    $ReduceTrustScore(S)$
  end if
  for $b \in B$ **do**
    $W'[b] = W'[b-1] \cup MiniBatchSGD(M'_i, W'_i)$
  end for
end if
Return W'_B

4.2.3 Defence at Federated Server:

Algorithm 3 illustrates the dual-layer defence and federated training process implemented at the S. Here, a continuous background broadcast of training requests, TRR_s, leads to the accumulation of interested devices, EDs. This accumulation process and broadcast continue to operate to ensure the required number of participants is available. In each communication round, a random subset of client devices, I, is chosen, and Algorithm 1 is executed in parallel to identify and remove malicious participants. In the initial communication round, M_0, W_0, and a shared secret key $K_{i,S}$ are established for each training participant. The management determines M_0 and W_0, while $K_{i,S}$ is established via a Diffie–Hellman exchange [133]. Only non-malicious clients, as determined by Algorithm 1, are permitted to train the model, and their weights are accumulated in $W_{i,r}$. At the conclusion of each communication round, the cosine similarity among client weights, X_S, is computed using Equation

Algorithm 3 Incorporated Defence at Federated Server, S

Input:
Initial Weights (W_0), Communication Rounds (R), Loss threshold (LTH), & Initial Model (M_0)
Output:
Global Model (M)

Broadcast training request, $TRR_{S,*}$
Run Parallel, $ED = getInterestedParticipants()$
for each $r \in R$ **do**
  Random Picked Clients, $I = randPickClientset(ED)$
  for $i \in I$ **do**
    if $r = 0$ **then**
      Shared Secret, $K_{i,S} = establishSharedSecret()$
      $W_{i,r-1} = W_0$
      $M_{i,r-1} = M_0$
    else
      if $isNotMalicious(i, K_{i,S})$ **then**
        $W_i, r \cup Train(W_{i,r-1}, M_{i,r-1}, K_{i,S})$
      end if
    end if
  end for
  Cosine Similarity Between Weights, $X_{S,I,r} = cosSim(W_{I,r})$
  $W_{I',r} = PickmWeights(W_{I,r}, X_{S,i,r})$
  $W^a_{I,r} = FedAVG(W_{I',r})$
  $M_{I,r} = Load(M_{I,r-1}, W^a_{I,r})$
  $L_{I,r} = ComputeLoss(W^a_{I,r}, M_{I,r})$
  if $L_{I,r} > LTH$ **then**
    $Discard(W^a_{I,r})$
  else
    $M = M \cup M_{I,r}$
  end if
end for

Return M

(6).

$$X_S[j,j'] = \frac{\boldsymbol{W}_j \cdot \boldsymbol{W}_{j'}}{\|\boldsymbol{W}_j\|\|\boldsymbol{W}_{j'}\|}, \quad \forall j,j' \in \{1,\ldots,n\},\ j \neq j'. \tag{6}$$

Subsequently, Equation (7) determines the weights $W_{I',r}$ for federated averaging (FEDAVG) to obtain W^a.

$$\boldsymbol{W}_{I',r} = \begin{cases} \left\{\boldsymbol{W}_j + \lambda \cdot \mathrm{sim}_j \cdot (\boldsymbol{W}_j - \boldsymbol{W}_{\mathrm{med}})\right\}_{j=1}^{n}, & \max(\mathrm{MAD}_j) \leq \gamma \\ \left\{\boldsymbol{W}_{(j)}\right\}_{j=1}^{m}, & \text{otherwise} \end{cases} \tag{7}$$

$$\begin{aligned} \text{where} \quad \mathrm{MAD}_j &= \operatorname*{med}_{k'} \left| X_S[j,k'] - \mathrm{med}(X_S) \right|, \\ \gamma &= \mathrm{med}(\mathrm{MAD}_{k'}) + \mathrm{Var}(\mathrm{MAD}_{k'}), \\ m &= \left\lfloor n \cdot \frac{\mathrm{med}(\mathrm{MAD}_{k'})}{\mathrm{med}(\mathrm{MAD}_{k'}) + \mathrm{Var}(\mathrm{MAD}_{k'})} \right\rfloor, \\ \mathrm{sim}_j &= \mathrm{med}_{k'}(X_S[j,k']), \\ \boldsymbol{W}_{\mathrm{med}} &= \text{element-wise median of } \{\boldsymbol{W}_j\}_{j=1}^{n} \end{aligned}$$

Under normal operating conditions (where $\max(\mathrm{MAD}_j) \leq \gamma$ for all clients $j \in n$), the weights are enhanced through a consensus-boosting operation: $W_j + \lambda \cdot \mathrm{sim}_j \cdot (W_j - W_{\mathrm{med}})$. Here, $\gamma = \mathrm{med}(\mathrm{MAD}_k) + \mathrm{Var}(\mathrm{MAD}_k)$ and $\mathrm{sim}_j = \mathrm{med}_k(X_S[j,k])$ represents client j's median cosine similarity. In the presence of adversarial conditions, the system selects $m = \left\lfloor n \cdot \frac{\mathrm{med}(\mathrm{MAD}_{k'})}{\mathrm{med}(\mathrm{MAD}_{k'}) + \mathrm{Var}(\mathrm{MAD}_{k'})} \right\rfloor$ weights by arranging all weights in ascending order according to their MAD_j values (with k' denoting the index of the weight vectors).

Subsequently, the averaged weights are loaded into the current model state $M_{I,r-1}$ to compute the loss $L_{I,r}$. A significantly high loss leads to the discarding of these model updates. The final global model M is achieved once the model reaches the desired level of accuracy.

4.2.4 Theoretical Analysis

This subsection provides theoretical proofs regarding the robustness and convergence properties of the proposed algorithms.

Theorem 1 (Robustness against Poisoning Attacks)**.** *The FedChallenger design is robust against poisoning attacks due to its challenge-response mechanism and malicious device detection algorithm. Specifically, the probability of a malicious participant maintaining a trust score T_i above the threshold T_{thresh} after d challenges is bounded by:*

$$P(T_i \geq T_{thresh}) \leq \left(\frac{1}{\alpha}\right)^{d-\log_\alpha(T_{init}/T_{thresh})},$$

where $\alpha > 1$ is the trust factor, T_{init} is the initial trust score, and T_{thresh} is the trust threshold.

Proof. The following mechanisms ensure the robustness of *FedChallenger*:

1. **Challenge-Response Mechanism:** Each participant i is challenged with a set of questions $Q = \{q_1, q_2, \ldots, q_n\}$. The expected response r_j for each challenge q_j is computed as $r_j = f(q_j)$, where $f(\cdot)$ is a deterministic function known only to legitimate devices. A malicious participant providing incorrect responses $r'_j \neq r_j$ is detected with high probability, as the probability of guessing all responses correctly is:

$$P_{\text{malicious}} = \prod_{j=1}^{n} P(r'_j = r_j) \leq \left(\frac{1}{|\mathcal{R}|}\right)^n,$$

where $|\mathcal{R}|$ is the size of the response space.

2. **Trust Score Dynamics:** The trust score T_i of participant i is updated as:

$$T_i \leftarrow \begin{cases} T_i \cdot \alpha, & \text{if } r'_j = r_j \quad \text{(correct response)}, \\ T_i/\alpha, & \text{if } r'_j \neq r_j \quad \text{(incorrect response)}, \end{cases}$$

where $\alpha > 1$ is the trust factor. Let T_{init} be the initial trust score. After l incorrect responses, the trust score becomes:

$$T_i = T_{\text{init}} \cdot \alpha^{-l}.$$

The participant is marked as malicious if $T_i < T_{\text{thresh}}$. Solving for l, we get:

$$l > \log_\alpha(T_{\text{init}}/T_{\text{thresh}}).$$

Thus, the number of incorrect responses required to mark the participant as malicious is:

$$l_{\text{required}} = \lceil \log_\alpha(T_{\text{init}}/T_{\text{thresh}}) \rceil.$$

The probability that a malicious participant provides fewer than l_{required} incorrect responses out of d challenges is bounded by:

$$P(T_i \geq T_{\text{thresh}}) \leq \left(\frac{1}{\alpha}\right)^{d-\log_\alpha(T_{\text{init}}/T_{\text{thresh}})}.$$

3. **Federated Aggregation with Cosine Similarity:** The server computes the cosine similarity matrix X_S for the weights $W = \{w_1, w_2, \ldots, w_n\}$ submitted by participants:

$$X_S[i,j] = \cos(w_i, w_j) = \frac{w_i \cdot w_j}{\|w_i\|\|w_j\|}.$$

The server filters out outliers by selecting m benign weights using $m = \left\lfloor n \cdot \frac{\text{med}(\text{MAD}_{k'})}{\text{med}(\text{MAD}_{k'}) + \text{Var}(\text{MAD}_{k'})} \right\rfloor$. Where, MAD_j is computed using

$$\text{MAD}_j = \underset{k'}{\text{med}} \left| X_S[j,k'] - \text{med}(X_S) \right|.$$

This ensures that only weights from non-malicious participants are included in the federated averaging process. Thus, the *FedChallenger* design is robust against poisoning attacks. ■

Theorem 2 (Convergence of FedChallenger)**.** *The FedChallenger design converges to a global model that minimizes the loss function, assuming that most participants are non-malicious and the learning rate is appropriately chosen.*

Proof. The following steps ensure the convergence of *FedChallenger*:

1. **Local Training Process:** Each non-malicious participant i performs local training using

mini-batch SGD. The updated rule for the weights w_i in next round denoted by $r+1$ is:

$$w_{i,r+1} = w_{i,r} - \eta_r \nabla \mathcal{L}_i(w_{i,r}),$$

where η_r is the learning rate and $\mathcal{L}_i(w)$ is the local loss function. Under standard assumptions (e.g., $\mathcal{L}_i(w)$ is Lipschitz smooth and convex) [57], mini-batch SGD converges to a local minimum of $\mathcal{L}_i(w)$.

2. **Federated Averaging:** The server aggregates the weights $W = \{w_1, w_2, \ldots, w_n\}$ from non-malicious participants using federated averaging:

$$w^a = \frac{1}{|W|} \sum_{i=1}^{|W|} w_i.$$

By the convexity of $\mathcal{L}_i(w)$, the federated average w^a satisfies:

$$\mathcal{L}(w^a) \leq \frac{1}{|W|} \sum_{i=1}^{|W|} \mathcal{L}_i(w_i),$$

where $\mathcal{L}(w)$ is the global loss function.

3. **Global Model Update:** The global model is updated iteratively as:

$$w_{r+1} = w_r - \eta_r \nabla \mathcal{L}(w_r).$$

Under the assumption that the majority of participants are non-malicious, the global model converges to a minimum of $\mathcal{L}(w)$.

4. **Loss-Based Filtering:** At each communication round, the server computes the loss $\mathcal{L}(w^a)$ and discards updates if $\mathcal{L}(w^a) > \mathcal{L}_{\text{thresh}}$, ensuring that only meaningful updates are applied to the global model.

Thus, the *FedChallenger* design ensures convergence to a global model that minimizes the loss function. ■

Lemma 1 (Trust Score Dynamics)**.** *The trust score T_i of a participant i decreases exponentially with*

the number of incorrect responses. Specifically, after l incorrect responses, the trust score becomes:

$$T_i = T_{init} \cdot \alpha^{-l},$$

where T_{init} is the initial trust score and $\alpha > 1$ is the trust factor.

Proof. The trust score T_i is updated as:

$$T_i \leftarrow T_i/\alpha \quad \text{(for each incorrect response)}.$$

After l incorrect responses, the trust score becomes:

$$T_i = T_{\text{init}} \cdot \alpha^{-l}.$$

Thus, the trust score decreases exponentially with the number of incorrect responses.

■

4.2.5 Security Analysis of Challenge-Response

The *FedChallenger* framework utilizes a mutual secret key ($K_{i,S}$)-based challenge-response mechanism to safeguard against poisoning attacks in FL. Therefore, the security of this challenge-response mechanism is fundamentally contingent on the confidentiality of $K_{i,S}$ and the unpredictability of the generated challenges. The encryption of challenges via $EC_{i,S} = K_{i,S}(c)$ offers information-theoretic security when $K_{i,S}$ has a key length of $\lambda \geq 128$ bits. The probability of compromising $K_{i,S}$ through a brute-force attack is given by Equation (8).

$$\Pr[\text{Compromise } K_{i,S}] \leq \varepsilon(\lambda) \approx 2^{-\lambda}, \tag{8}$$

where $\varepsilon(\lambda)$ signifies the negligible success probability.

Furthermore, the unpredictability of the challenge-response is linked to the random generation of $CS \in CM$. Here, CM integrates multiple sources of entropy, including the device's static,

dynamic, and ephemeral characteristics. Static traits encompass, but are not limited to, device fingerprints and historical patterns. Dynamic characteristics, conversely, include recent activity logs and temporal usage patterns. Ephemeral information comprises session-specific nonces. The probability that an attacker correctly guesses the i-th challenge, c_i, given their knowledge (A) about the participant, is provided by Equation (9).

$$\Pr[\text{Successful guess}] = \prod_{i=1}^{n} \Pr[c_i \in CS | \mathcal{A}]. \tag{9}$$

Moreover, without knowledge of $K_{i,S}$, the probability is $\Pr[c_i \in CS] \approx \frac{1}{|CM|}$. Even if the attacker possesses partial knowledge of CM, the probability $\Pr[c_i \in CS]$ remains negligible. Additionally, the trust score mechanism provides adaptive protection. Consequently, the challenges and responses offer sufficient security for most practical scenarios.

4.3 Experimental Configuration

This section describes the experimental framework designed to assess the *FedChallenger* method's effectiveness in countering model and data poisoning attacks. All experiments were performed on a MacBook M1 Pro workstation featuring an 8-core processor, 14-core graphics unit, 16GB of memory, and 512GB of solid-state storage. The selected neural architecture was ResNet-18, which initiates with a 7×7 convolution layer using 64 filters at a stride of two with 'same' padding, followed by batch normalization, ReLU nonlinearity, and 3×3 maximum pooling. The network structure continues with four consecutive residual modules: the initial pair utilizes two 3×3 convolutional layers with 64 and 128 filters correspondingly, each preceded by batch normalization and ReLU, while the subsequent modules scale up to 256 and 512 filters, maintaining identical kernel dimensions, integrating skip pathways to address gradient dissipation. Final feature representations undergo global average pooling before passing through a classification layer consisting of one linear transformation with softmax output for multi-category prediction. The optimization process utilizes mini-batch SGD with a 0.001 learning rate and 20 samples per batch, performing five local training cycles per participant across 100 clients with non-identical data distributions.

Table 4.1: Data Partitioning Statistics

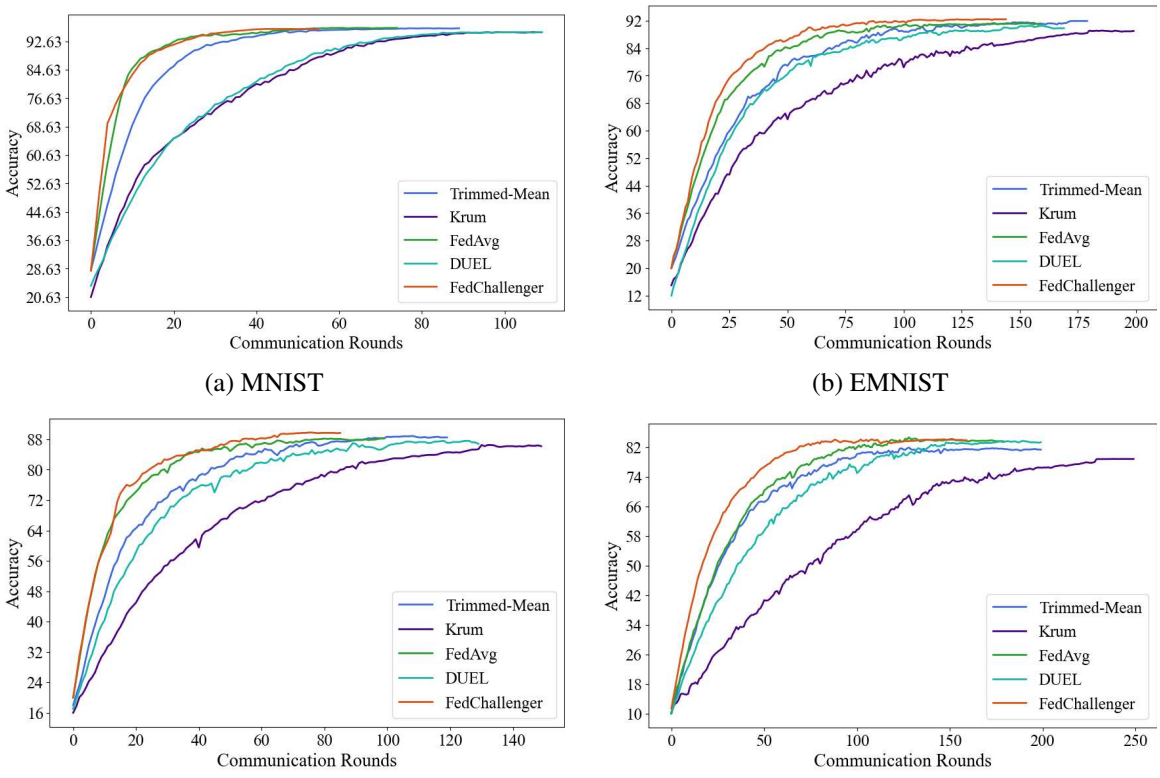
Dataset	Total Examples	Training	Testing	Validation
MNIST	60,000	45,000	10,000	5,000
FMNIST	60,000	45,000	10,000	5,000
CIFAR-10	60,000	45,000	10,000	5,000
EMNIST	382705	3,00,000	50,832	31,873

The heterogeneous data evaluation environment was established using MNIST, FMNIST, EMNIST, and CIFAR-10 datasets following [135] for thorough dataset-specific analysis. These four benchmark collections were selected due to their varying complexity levels and established status in FL experimentation. MNIST offers elementary grayscale numeral recognition, whereas FMNIST raises difficulty with apparel classification at identical dimensions. EMNIST broadens the scope by incorporating handwritten characters alongside digits. CIFAR-10 presents additional challenges with colored imagery and intricate object identification. Collectively, these datasets facilitate comprehensive assessment across diverse data configurations and problem complexities, spanning from elementary pattern recognition to advanced visual categorization. Their consistent formats and prevalent adoption in FL studies guarantee fair comparisons with recent techniques while encompassing the crucial range of non-IID data obstacles. Table 4.1 displays the data distribution where MNIST, FMNIST, and CIFAR-10 each contain 60,000 instances, with 45,000 allocated for model training, 10,000 for testing, and 5,000 for validation. The EMNIST collection utilizes 300,000 training examples, 50,832 test cases, and 31,873 validation samples. During non-IID data preparation, each participant received data from exactly two categories according to McMahan *et al.*'s [80] methodology. Additionally, the error threshold served as the stopping condition for the experiments. For model poisoning, we implemented the widely-used Gaussian noise injection [141] where the j^{th} weight is substituted with a value sampled from a normal distribution. Regarding data poisoning simulation, we employed the extensively-studied label replacement approach [50], randomly swapping the original class label with other valid labels from the dataset.

FedChallenger's assessment was performed under three conditions: i) no-attack, ii) Model Poisoning attack, and iii) Data Poisoning attack. The Gaussian noise method was applied for model

parameter tampering, with *FedChallenger*'s prediction accuracy recorded across varying proportions of affected nodes. In the absence of attacks, the accuracy was assessed to underscore the robustness of the proposed methodology. Additionally, measurements of the average convergence time underlined the efficacy of *FedChallenger* under benign conditions. When subjected to a data poisoning attack, model accuracy was quantified across varying percentages of data Label Flipping. The proportion of compromised devices present in the system directly influenced the extent of Label Flipping. To establish *FedChallenger*'s advantages, these evaluation conditions included comparisons against DUEL [34], Trimmed-Mean[143], Krum [15], FedAvg [79], Stake [31], Shap[56], and Cluster [48] methodologies.

4.4 Results and Discussions



(a) MNIST (b) EMNIST

(c) FMNIST (d) CIFAR-10

Figure 4.2: No Attack Scenario: Evaluation of Accuracy.

This section delves into the performance evaluation of the *FedChallenger* technique, contrasting its effectiveness with several established approaches: FedAvg, Krum, Trimmed-Mean, DUEL, Stake, Shap, and Cluster. The comprehensive evaluation is organized into three distinct scenarios: no-attack scenario, model poisoning scenario, and data poisoning attack scenario.

4.4.1 No-Attack Scenario:

Fig. 4.2 illustrates a comparative assessment of the test accuracy achieved by *FedChallenger* against several established baseline methodologies, including FedAvg, Krum, Trimmed-Mean, and the DUEL approach. This comparison spans the MNIST, FMNIST, EMNIST, and CIFAR-10 datasets, all evaluated under benign conditions. *FedChallenger*'s novel aggregation mechanism, which heavily relies on cosine similarity, allows it to attain competitive accuracy using fewer communication rounds. Concurrently, it offers enhanced stability, a characteristic especially beneficial within non-IID settings.

During the MNIST dataset evaluation, *FedChallenger*'s performance proved to be on par with that of both FedAvg and Trimmed-Mean, consistently exceeding 96% accuracy within the initial training rounds. Conversely, Krum's method and the DUEL approach required additional communication rounds to reach their peak accuracy of 95%. This observed difference in performance primarily arises from Krum's inherent constraint of exclusively relying on a single client update per round, which severely limits the diversity of gradients considered. Meanwhile, the DUEL approach encounters computational inefficiencies primarily due to its reliance on dual rejection mechanisms, which integrate both Loss Function Rejection (LFR) and Error Rate Rejection (ERR). *FedChallenger*'s comparable performance in this context is attributable to its strategy of consensus boosting updates. This approach ensures that updates maintaining strong directional alignment with the global model, as determined by cosine similarity, contribute effectively, thus promoting stable convergence while diligently preserving all beneficial updates.

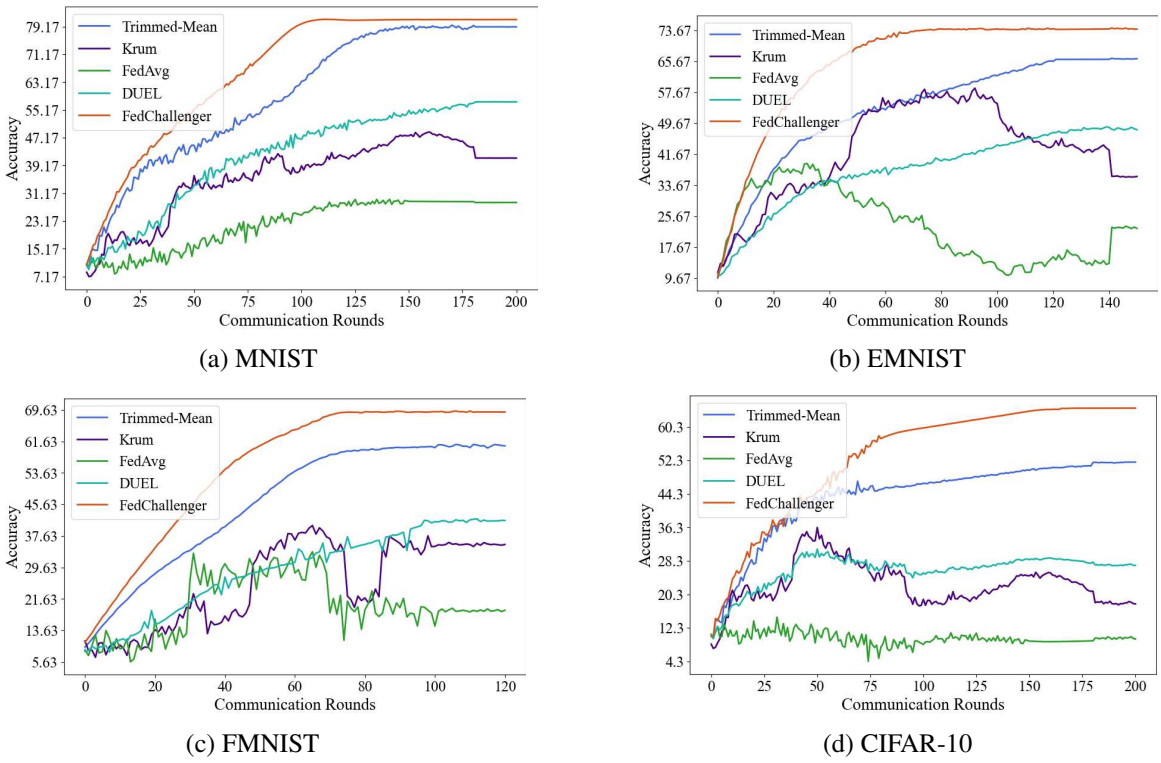
For the FMNIST dataset evaluations, *FedChallenger* clearly established a significant accuracy advantage, outperforming Trimmed-Mean by 1.5%, Krum by 4.1%, FedAvg by 1.3%, and the DUEL approach by 3.4%. Simultaneously, it demonstrated notably steeper initial learning curves. Krum, Trimmed-Mean, and the DUEL approach exhibited a slower capacity to adapt to FMNIST's

inherently complex feature space. In contrast, *FedChallenger*'s similarity-driven weighting mechanism effectively managed to balance contributions across a diverse array of clients. This particular capability proved exceptionally valuable in mitigating noise that arises from non-IID data distributions. The comparative strategies of Krum and DUEL exhibited notably flatter learning curves, a consequence of their respective limitations—Krum's excessive rigidity in its update selection process and the DUEL approach's tendency towards unnecessarily discarding updates.

The EMNIST dataset evaluation further demonstrates *FedChallenger* maintaining performance equivalency with FedAvg and Trimmed-Mean, while concurrently achieving considerable accuracy improvements of 3.9% over Krum and 2.9% over the DUEL approach. *FedChallenger*'s exceptional stability, particularly under non-IID conditions, is directly derived from its consensus-boosted weights and the rigorous enforcement of gradient alignment through a cosine similarity matrix. This methodological approach effectively averts destabilisation that might otherwise result from skewed local updates. Such destabilisation is a vulnerability particularly evident in methods like FedAvg and Trimmed-Mean, given their inherent lack of explicit mechanisms for verifying geometric consistency.

Within the CIFAR-10 dataset experiments, *FedChallenger* consistently matched the accuracy levels exhibited by FedAvg and the DUEL approach, while substantially surpassing Krum by 6.4% and Trimmed-Mean by 3.1%. The observation of early convergence in Fig. 4.2 distinctly highlights *FedChallenger*'s pronounced efficiency, mainly when operating within high-dimensional data spaces. In such environments, Krum's similarity-based selection criteria often prove inadequate in maintaining sufficient gradient diversity. Trimmed-Mean, conversely, struggles with the inherent properties of the CIFAR-10 dataset and its reliance on a fixed threshold-based rejection methodology. In contrast, *FedChallenger*'s lightweight cosine-based weighting approach successfully maintains both scalability and robust performance.

A pivotal observation drawn from the analysis across all datasets reveals that *FedChallenger* exhibits a markedly reduced degree of accuracy fluctuation under non-IID conditions when contrasted with the baseline methods, as comprehensively illustrated in Fig. 4.2. This substantial advantage in stability is a direct consequence of *FedChallenger*'s gradient alignment enforcement strategy, which intelligently weights client updates based on their directional consistency. This methodical approach



(a) MNIST (b) EMNIST

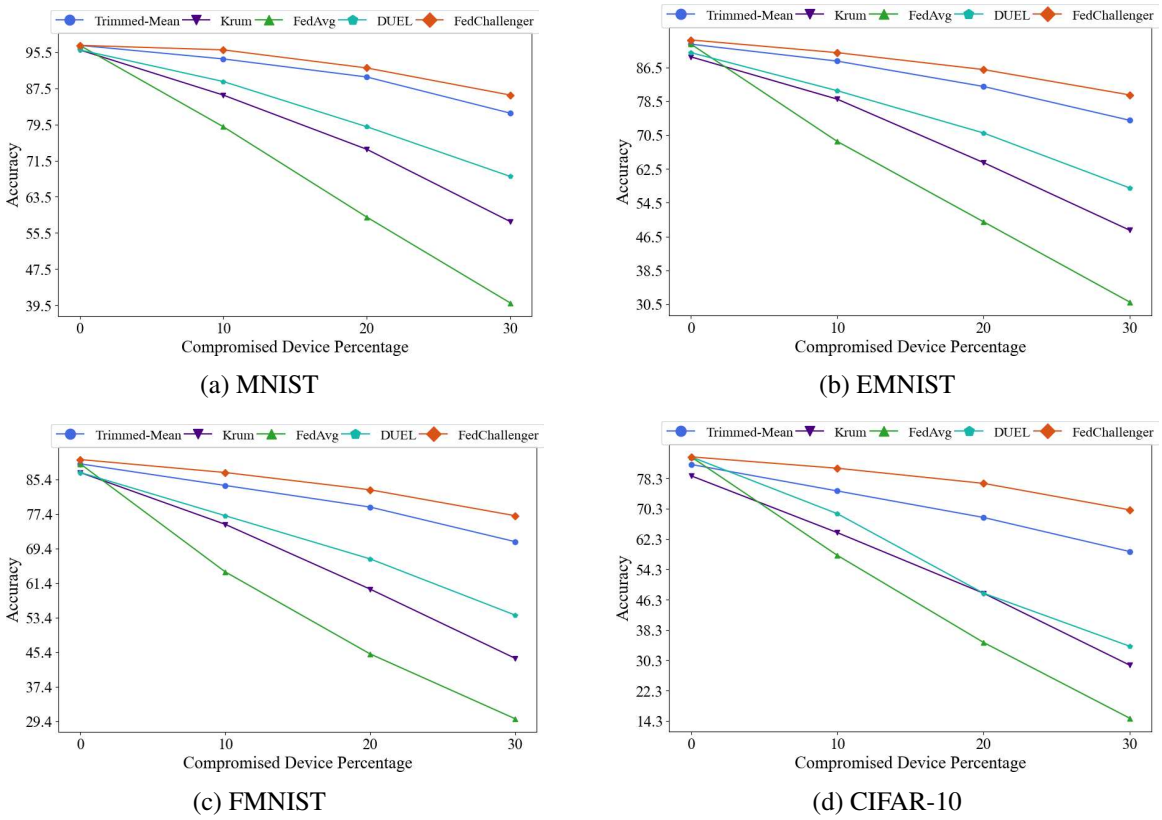
(c) FMNIST (d) CIFAR-10

Figure 4.3: Model Poisoning Attack Scenario: Evaluation of Accuracy under 40% Compromised Devices.

effectively mitigates the issue of client drift without resorting to the aggressive pruning characteristic of Trimmed-Mean or Krum's problematic over-reliance on single updates. In contrast, FedAvg's straightforward uniform averaging inevitably aggregates variance stemming from divergent client contributions. In contrast, the DUEL approach's rejection-based methodology often unnecessarily discards potentially valuable updates even in benign operational environments. Furthermore, *FedChallenger* incorporates MAD-based adaptive pruning of gradients, a design feature that guarantees zero filtered records in the context of the no-attack scenario.

4.4.2 Model Poisoning Attack Scenario

In the model poisoning attack scenario, adversarial actors substitute the target device's model parameters with values sampled from a Gaussian distribution. Fig. 4.3 presents the accuracy comparison between *FedChallenger*, Krum, Trimmed-Mean, FedAvg, and DUEL approaches throughout consecutive iterations under this attack condition. The experiment examines the substitution of



(a) MNIST

(b) EMNIST

(c) FMNIST

(d) CIFAR-10

Figure 4.4: Model Poisoning Attack Scenario: Evaluation of Accuracy under Different Compromised Device Percentages.

40% of clients' model parameters with Gaussian-distributed values, with experiments performed on MNIST, FMNIST, EMNIST, and CIFAR-10 datasets to validate *FedChallenger*'s resilience.

The findings show notable accuracy variations in Krum and FedAvg, establishing them as the poorest-performing techniques. *FedChallenger* exhibits considerable accuracy enhancements across all datasets, attaining 2.9, 3.2, 3.68, and 6.7 times superior accuracy compared to FedAvg on MNIST, EMNIST, FMNIST, and CIFAR-10, respectively. Relative to the DUEL approach, *FedChallenger* sustains accuracy improvements of 1.41, 1.54, 1.66, and 2.38 times across the identical datasets. Additionally, *FedChallenger* achieves absolute accuracy enhancements of 2.65%, 14.24%, 11.15%, and 24.85% over Trimmed-Mean on MNIST, FMNIST, EMNIST, and CIFAR-10, respectively. Krum's Euclidean distance-based intensive outlier elimination shows particular susceptibility, producing 1.9, 2.6, 1.4, and 3.6 times reduced accuracy compared to *FedChallenger* on the

corresponding datasets. The evaluation definitively establishes *FedChallenger*'s reliable superiority across all datasets. Its adaptive m update selection, utilizing MAD-based estimation, facilitates dynamic pruning of updates while guaranteeing cosine similarity emphasizes update direction over absolute magnitudes. This methodology renders *FedChallenger* resistant to Gaussian noise and distorted gradients resulting from non-IID data.

The evaluation framework examines *FedChallenger*'s resilience across varying percentages of compromised devices. Fig. 4.4 displays the accuracy patterns from 0% to 30% poisoning rates, where malicious clients introduce Gaussian-distributed noise into model parameters. Aligned with theoretical predictions, all baseline techniques exhibit progressive accuracy deterioration as poisoned updates escalate.

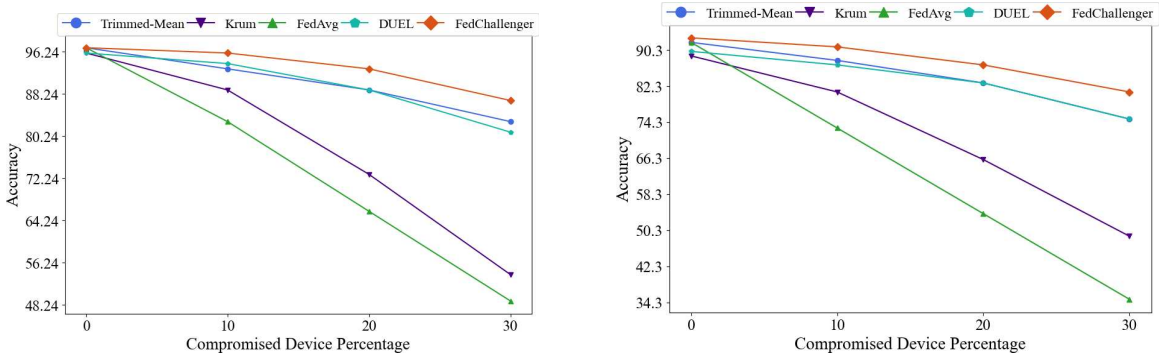
FedChallenger sustains outstanding performance across all datasets, obtaining accuracy improvements of 2.2, 2.6, 2.5, and 4.9 times compared to FedAvg in MNIST, EMNIST, FMNIST, and CIFAR-10 evaluations. Relative to the DUEL approach, *FedChallenger* demonstrates comparative accuracy gains of 23.6%, 32.8%, 35.4%, and 69.7% across the identical datasets.

Although Trimmed-Mean appears as the most formidable competitor, *FedChallenger* preserves steady advantages of 5.2%, 8.3%, 7.8%, and 17.2% in MNIST, FMNIST, EMNIST, and CIFAR-10 evaluations, respectively. This performance benefit originates from *FedChallenger*'s sophisticated aggregation strategy, which extends Trimmed-Mean by integrating cosine similarity for improved robustness. Particularly, it dynamically eliminates updates through MAD computation, modifying the m value to strengthen resilience additionally.

Krum's core constraint of selecting merely a single client update per aggregation round proves especially damaging, causing severe accuracy deficits of 40.1%, 55.2%, 51.2%, and 83.8% compared to *FedChallenger* across FMNIST, EMNIST, and CIFAR-10 evaluations. These thorough experimental findings conclusively establish *FedChallenger*'s superior accuracy and robustness across all examined scenarios and datasets.

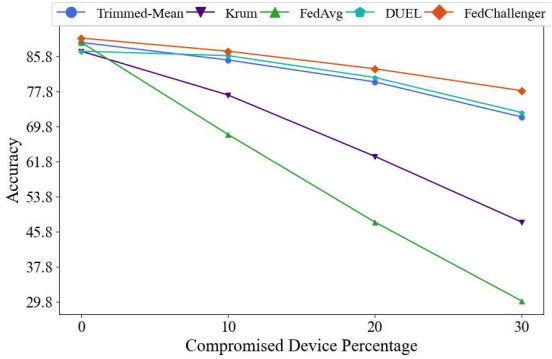
4.4.3 Data Poisoning Attack Scenario:

In this experiment, the target label within the training set is substituted with a randomly chosen label from the available list, ensuring the current label is excluded. Fig. 4.5 visually depicts

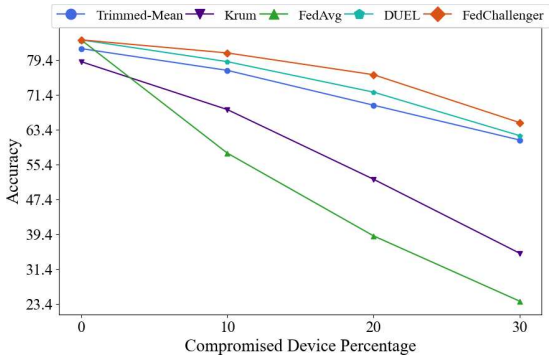


(a) MNIST

(b) EMNIST

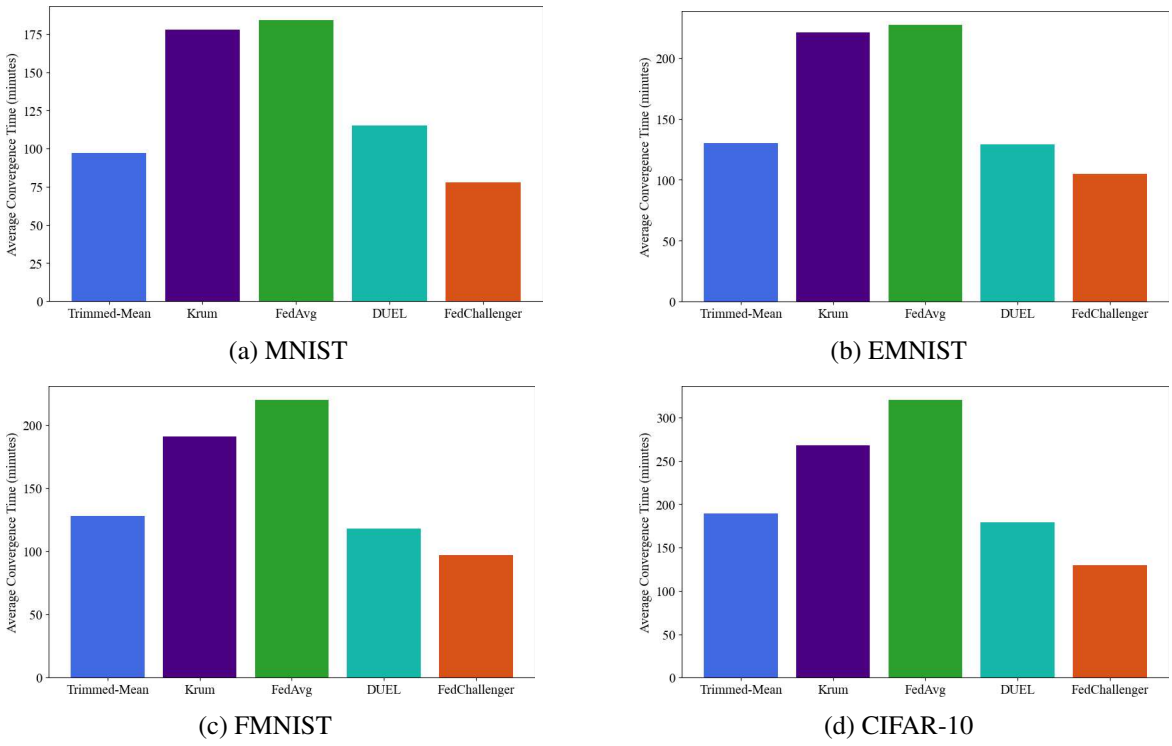


(c) FMNIST



(d) CIFAR-10

Figure 4.5: Data Poisoning Attack Scenario: Evaluation of Accuracy under Different Compromised Device Percentages.



(a) MNIST (b) EMNIST

(c) FMNIST (d) CIFAR-10

Figure 4.6: 20% Data Poisoning Attack Scenario: Evaluation of Average Convergence Time.

the data poisoning attack scenarios, with a primary focus on random Label Flipping. This evaluation examines cases where 0% to 30% of the clients' labels undergo flipping. The obtained results clearly demonstrate that *FedChallenger* consistently outperforms other established methods. As the percentage of Label Flipping increases, the accuracy of Trimmed-Mean experiences a significant decline, specifically, under a scenario involving 30% compromized devices, Trimmed-Mean experiences an accuracy drop of 4.6% on MNIST, 7.6% on EMNIST, 8.2% on FMNIST, and 7.2% on CIFAR-10. Similarly, while Krum maintains consistency in model poisoning and no-attack scenarios, its performance becomes unstable as Label Flipping intensifies. *FedChallenger* achieves considerably higher accuracy than Krum, with improvements reaching up to 47.2% on MNIST, 61.5% on CIFAR-10, 47.8% on FMNIST, and 49.7% on EMNIST when 30% of device labels are flipped.

Furthermore, when compared to *FedChallenger*, both FedAvg and the DUEL approach exhibit substantial accuracy degradation under 30% Label Flipping. FedAvg endures drops of 93.5% on CIFAR-10, 56.4% on MNIST, 88.4% on FMNIST, and 80.4% on EMNIST, whereas the DUEL

method shows declines of 5.6% on CIFAR-10, 7.1% on MNIST, 7.4% on FMNIST, and 7.8% on EMNIST.

Despite 30% of data labels being flipped, *FedChallenger* consistently maintains superior performance, demonstrating significantly higher accuracy than competing methods. These findings confirm that *FedChallenger* exhibits robustness against both model and data poisoning attacks. This resilience is directly attributable to its challenge-response-based mechanism and its cosine similarity-based weight selection. The challenge-response mechanism dynamically verifies the credibility of participant updates, and cosine similarity coupled with adaptive m-value pruning utilizing MAD, effectively eliminates inconsistent updates.

Convergence Guarantee: Beyond enhancing accuracy, *FedChallenger* guarantees the most rapid convergence among the compared baseline approaches across varying intensities of model and data poisoning attacks. Fig. 4.6 provides a comparative analysis of the average convergence times for *FedChallenger*, Trimmed-Mean, Krum, FedAvg, and the DUEL approach, evaluated on MNIST, EMNIST, FMNIST, and CIFAR-10 datasets when 20% of client data is poisoned.

On the MNIST dataset, *FedChallenger* achieves a 19.5% reduction in convergence time when compared to Trimmed-Mean and outperforms the DUEL method by 32.2%. This performance advantage becomes even more pronounced when benchmarked against FedAvg and Krum, showcasing improvements in convergence time of 57.1% and 56.2%, respectively. This substantial disparity in performance originates from *FedChallenger*'s highly efficient cosine similarity-based aggregation mechanism, which effectively filters malicious updates while diligently preserving valid gradient contributions.

The FMNIST evaluation yields similar outcomes, with *FedChallenger* converging 24.2% faster than Trimmed-Mean and 17.8% faster than the DUEL approach. The baseline methods of FedAvg and Krum demonstrate notably slower convergence, requiring 55.9% and 49.4% more time, respectively, to reach comparable accuracy levels.

EMNIST experiments further validate *FedChallenger*'s robustness, showing convergence time improvements of 53.7% over FedAvg and 52.5% over Krum. When compared to more sophisticated approaches, *FedChallenger* maintains a 19.2% advantage over Trimmed-Mean and an 18.6% improvement over the DUEL method.

The consistent performance observed across diverse datasets emphasizes the effectiveness of *FedChallenger*'s aggregation strategy in managing varying data complexities. The most challenging CIFAR-10 evaluation reveals *FedChallenger*'s strongest performance advantages, particularly against vulnerable baseline methods. *FedChallenger* converges 60.0% faster than FedAvg and 52.0% faster than Krum. When compared to more robust approaches, it maintains significant leads of 31.2% over Trimmed-Mean and 27.4% over DUEL. This definitively demonstrates *FedChallenger*'s scalability to complex, high-dimensional data spaces while preserving its convergence advantages.

The comprehensive evaluation conducted across MNIST, FMNIST, EMNIST, and CIFAR-10 demonstrates that *FedChallenger* achieves consistent convergence time improvements ranging from 17.8% to 60.0% when compared to existing approaches. These results conclusively indicate that *FedChallenger*'s background challenge-response and lightweight cosine similarity-based aggregation utilizing MAD computation collectively achieve both faster convergence and enhanced robustness against poisoning attacks across diverse datasets and complexity levels.

4.4.4 Performance Comparison with Contemporary Methods

To assess *FedChallenger*'s effectiveness relative to state-of-the-art defence mechanisms—namely, Stake [31], Shap [56], and Cluster [50]—against poisoning attacks, a comprehensive evaluation was performed for both data and model poisoning cases.

The Stake framework implements a blockchain-driven consensus protocol incorporating participant incentive mechanisms. During evaluation, Stake's functionality was replicated through an Ethereum smart contract setup [58] with $\varepsilon = 0.5$ using Web3.py [31]. Shap employs SHAP (Shapley Additive exPlanations) values for poisoning attack detection, where 10% of testing instances formed the reference dataset for SHAP calculations. The Cluster method analyzes gradient updates from participating clients to distinguish source and target categories, followed by HDBSCAN clustering [142] to isolate malicious contributions.

In model poisoning simulations, network parameters from randomly chosen clients were substituted with values drawn from a Gaussian distribution. For Label-Flipping attacks, original training labels were randomly swapped with alternative class identifiers. The experimental configuration

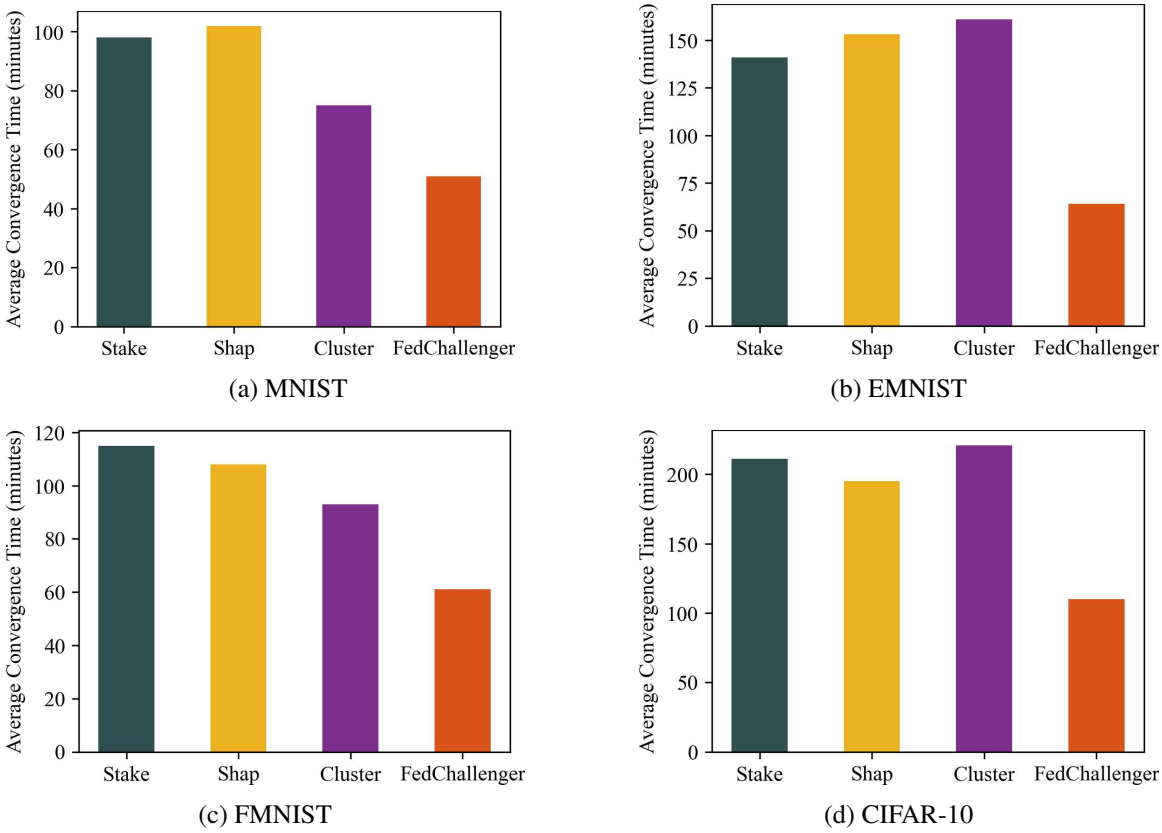


Figure 4.7: No Attack Scenario: Evaluation of Average Convergence Time.

matched previous evaluations to maintain consistency. Performance metrics encompass both model convergence time and classification accuracy under data poisoning and model poisoning attack conditions.

Convergence Time Guarantee: Fig. 4.7 displays the mean convergence duration for Stake, Shap, Cluster, and *FedChallenger* methods under no-attack conditions, demonstrating their efficiency in standard operational settings. The results indicate that dataset characteristics and size influence model convergence rates.

During MNIST testing, Shap requires approximately twice the convergence time of *FedChallenger* due to computational overhead from SHAP value derivation. At the same time, Stake exhibits nearly 63% longer convergence owing to blockchain-related computations. Cluster employs HDBSCAN clustering, resulting in roughly 38% slower convergence compared to *FedChallenger*. *FedChallenger* implements an efficient challenge-response protocol alongside cosine similarity-driven consensus enhancement before aggregation, achieving substantially reduced computational overhead relative to Stake, Shap, and cluster-based approaches.

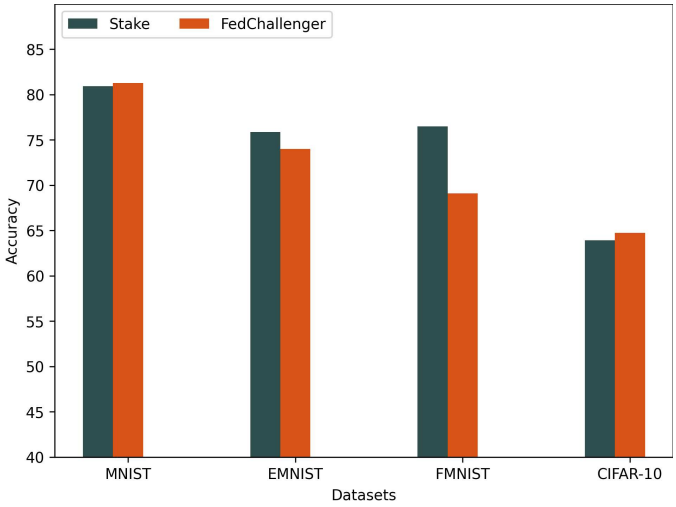
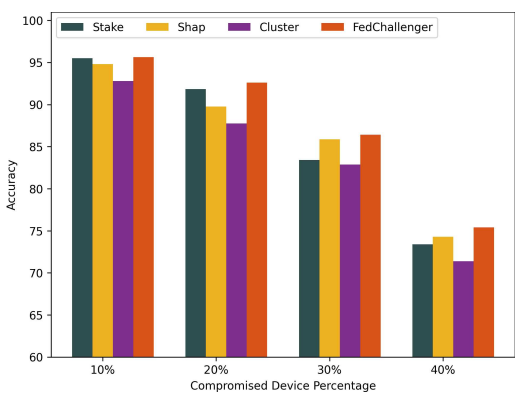
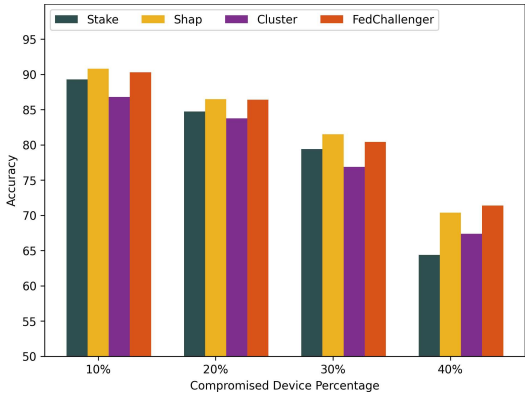


Figure 4.8: Model Poisoning Attack Scenario: Evaluation of Accuracy Under 40% Compromised Device Scenario.

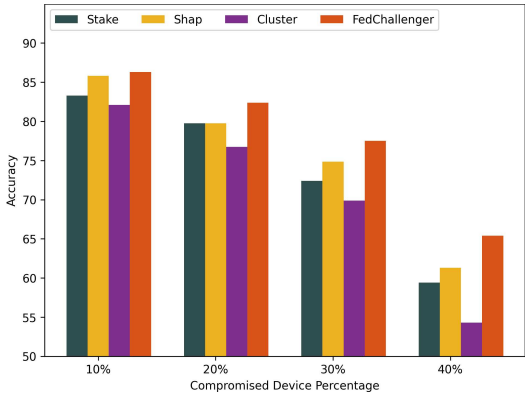
Extended evaluation using EMNIST, FMNIST, and CIFAR-10 datasets demonstrates *FedChallenger*'s superior computational efficiency compared to recent methods. For EMNIST, *FedChallenger* achieves convergence 2.2, 2.4, and 2.5 times faster than Stake, Shap, and Cluster techniques,



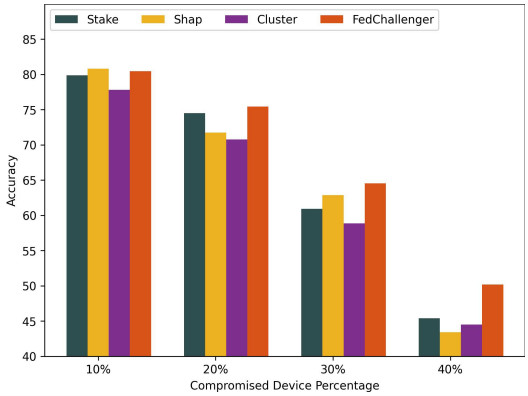
(a) MNIST



(b) EMNIST



(c) FMNIST



(d) CIFAR-10

Figure 4.9: Data Poisoning Attack: Evaluation of Accuracy under Different Compromised Device Percentage.

respectively. Similarly, in FMNIST experiments, *FedChallenger* maintains performance advantages of 1.8, 1.6, and 1.4 times over Stake, Shap, and Cluster approaches, correspondingly.

The CIFAR-10 evaluation demonstrates *FedChallenger*'s superior efficiency, achieving 46%, 42%, and 50% faster convergence compared to Stake, Shap, and Cluster techniques, respectively. These findings validate *FedChallenger* as a computationally efficient alternative to contemporary state-of-the-art defence mechanisms.

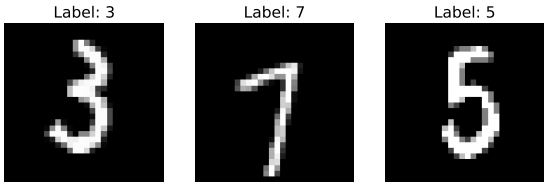
Model Poisoning Attack: This experiment provides a thorough assessment of model poisoning resistance by systematically analyzing *FedChallenger* and Stake aggregation approaches across MNIST, EMNIST, FMNIST, and CIFAR-10 datasets. Other defence methods were omitted from this analysis as they primarily target data poisoning rather than model poisoning scenarios. The experimental setup assumes adversarial manipulation affecting 40% of client model parameters.

The outcomes, illustrated in Fig. 4.8, reveal distinct performance patterns between the two defence strategies. *FedChallenger* demonstrates enhanced robustness in MNIST and CIFAR-10 environments, delivering accuracy gains of 0.5% and 1.1% respectively over Stake. However, Stake performs marginally better on EMNIST and FMNIST datasets, with accuracy improvements of 2.4% and 9.5% respectively. These variations suggest dataset-specific effectiveness patterns, with *FedChallenger* showing powerful generalization capabilities in complex CIFAR-10 environments.

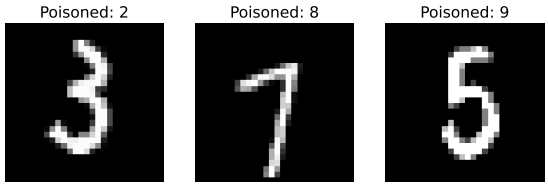
Data Poisoning Attack Evaluation: The data poisoning scenario involved random substitution of class labels with distinct alternatives. Testing spanned MNIST, EMNIST, FMNIST, and CIFAR-10 datasets with compromised device percentages ranging from 10% to 40%. Fig. 4.9 displays the attack outcomes across different compromise levels and datasets.

For MNIST, *FedChallenger* shows minimal accuracy improvement over Shap, while Stake and Cluster exhibit accuracy reductions of approximately 3% and 6% respectively at 40% compromised device scenario. EMNIST evaluation reveals Cluster suffering a 6% accuracy deficit compared to *FedChallenger* at 40% compromised device scenario, with Shap showing a 1.5% decrease. Stake performs poorest in this scenario, trailing *FedChallenger* by over 10% accuracy.

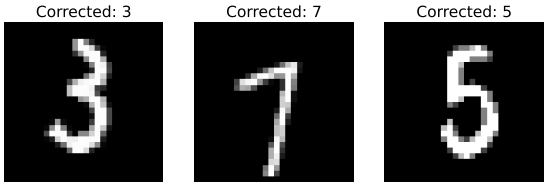
FMNIST evaluation at 40% compromised device scenario shows Cluster as the weakest performer with a 20% accuracy deficit relative to *FedChallenger*, while Stake and Shap lag by 9.5%



(a) Original samples (labels: 3, 7, 5)



(b) Poisoned samples (flipped to: 2, 8, 9)



(c) FedChallenger corrections

Figure 4.10: Demonstration of Label Flipping attack and mitigation on MNIST samples.

and 6% respectively. CIFAR-10 analysis demonstrates greater accuracy degradation across all methods compared to *FedChallenger*, with Shap showing abysmal performance (15% accuracy loss), followed by Stake (10%) and Cluster (12%).

The evaluation confirms *FedChallenger*'s effectiveness, attributable to its challenge-response protocol and enhanced Trimmed-Mean aggregation incorporating MAD-based adaptive m value pruning with cosine-similarity analysis. Results demonstrate consistent robustness across diverse datasets and faster convergence enabled by the method's computationally efficient algorithms.

4.4.5 Demonstration with Exemplary MNIST Data

An empirical validation of the *FedChallenger* methodology was conducted using a subset of the MNIST dataset subjected to Label Flipping attacks. Figure 4.10 illustrates both clean and poisoned samples alongside *FedChallenger*'s corrective process. The unaltered samples correctly display digits 3, 7, and 5, while the compromised versions are mislabeled as 2, 8, and 9, respectively. *FedChallenger* successfully restores the original labels (3, 7, and 5) through its mitigation process.

Table 4.2 provides quantitative analysis of MNIST classification performance under 30% Label Flipping attacks. The experimental validation demonstrates *FedChallenger* achieving 86.7% classification accuracy under Label Flipping attacks, surpassing baseline approaches by margins ranging from 4% to 57%. Figure 4.11 visualizes the cosine similarity mechanism for detecting malicious client updates, clearly showing successful identification of poisoned updates (similarity

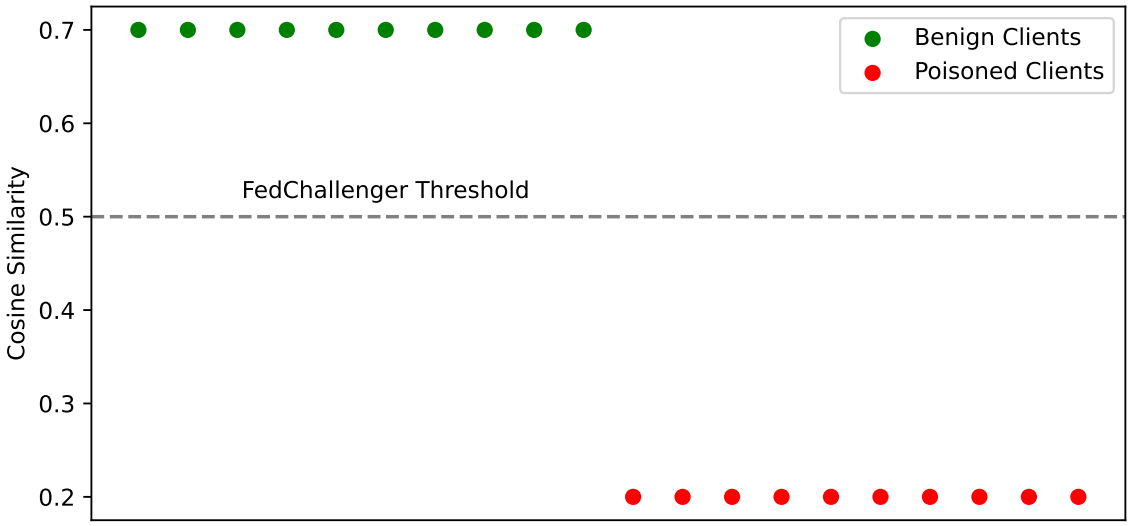


Figure 4.11: Gradient cosine similarity between benign and poisoned clients.

Table 4.2: Accuracy comparison with 30% Label Flipping on MNIST

Method	Accuracy (%)
FedAvg	48.2 ± 1.4
Krum	53.7 ± 2.1
Trimmed-Mean	83.5 ± 1.8
FedChallenger	$\mathbf{86.7 \pm 0.9}$

scores < 0.3) while preserving legitimate updates (similarity scores > 0.7).

4.4.6 Ablation Study

To examine the effects of adversarial conditions and parameter configurations, we perform an ablation study.

Impact of Compromised Device Percentages

Table 4.3 details our ablation study evaluating the state-of-the-art techniques' performance across different proportions of compromised devices with poisoned data. The analysis employs the following experimental configuration:

- **Datasets**: Two widely used datasets, EMNIST and CIFAR-10, are employed to evaluate the techniques.

Table 4.3: F1-Score for Techniques Under Different Percentage of Compromised Devices.

Techniques	$\alpha = 0$		$\alpha = 10$		$\alpha = 20$		$\alpha = 30$	
	EMNIST	CIFAR-10	EMNIST	CIFAR-10	EMNIST	CIFAR-10	EMNIST	CIFAR-10
Trimmed-Mean	91.34	83.93	87.56	76.35	82.54	68.55	74.52	60.37
Krum	89.25	78.67	80.34	67.54	65.81	51.46	48.24	34.22
FedAvg	91.84	83.34	72.33	57.38	53.17	38.48	34.35	23.46
DUEL	90.26	81.43	78.53	78.39	82.47	71.54	74.37	61.33
Stake	91.94	82.94	89.39	79.86	84.75	74.50	79.47	63.91
Shap	91.19	**84.14**	90.86	80.83	86.53	73.75	**81.58**	64.87
Cluster	90.54	81.54	86.85	77.83	83.75	70.75	76.87	60.87
FedChallenger	**92.48**	83.98	**91.37**	**81.45**	**86.74**	**75.45**	80.4	**65.53**

- **Adversarial Conditions**: The percentage of compromised devices α is varied at 0%, 10%, 20%, and 30% to simulate increasing levels of adversarial influence.
- **Metric**: The F1-Score is used to measure the performance of each technique.

The findings of the ablation study are highlighted as follows:

- **Baseline Performance** ($\alpha = 0$): All techniques perform well without adversarial influence. *FedChallenger* achieves the highest F1-Score of 92.48% on the EMNIST dataset, while the Shap technique achieves the highest F1-Score of 84.14% on CIFAR-10. Stake remains a close contender on EMNIST with an F1-Score of 91.94%, and *FedChallenger* ranks second on CIFAR-10 with an F1-Score of 83.98%.
- **Impact of Adversarial Conditions** ($\alpha > 0$): As the percentage of compromised devices increases, the performance of most techniques degrades. *FedChallenger* maintains the highest F1-Score across most levels of α, demonstrating its robustness. However, it achieves the second-highest F1-Score on the EMNIST dataset when 30% of devices are compromised, with Shap emerging as the clear winner. In contrast, FedAvg shows significant vulnerability, with its F1-Score dropping sharply from 83.34% at $\alpha = 0$ to 23.46% at $\alpha = 30$ on the CIFAR-10 dataset. Krum is the second most vulnerable strategy, with F1-Scores of 48.24% and 34.22% in the 30% compromised device scenario on the EMNIST and CIFAR-10 datasets, respectively.

- **Comparative Analysis**: *FedChallenger* consistently outperforms other techniques, highlighting its effectiveness in adversarial conditions. Techniques like Stake, Shap, and Cluster exhibit moderate robustness, while FedAvg and Krum are the most vulnerable to adversarial influence.

Impact of Batch Size and Learning Rate

This experiment evaluates the effects of batch size (B) and learning rate (η) variations exclusively through F1-Score measurements on the CIFAR-10 dataset, with a fixed $\alpha = 20$. The outcomes are presented in Table 4.4 where the

Table 4.4: F1-Scores (%) under varying B and η.

Techniques	$B = 10$ $\eta = 0.001$	$B = 20$ $\eta = 0.001$	$B = 20$ $\eta = 0.01$
FedAvg	36.54	38.48	40.69
Krum	48.37	51.46	55.64
FedChallenger	**83.15**	**75.45**	**80.11**

Key Findings are:

- **Batch Size**: Larger batches ($B = 20$) generally improve robustness (e.g., FedAvg gains +1.94% over $B = 10$).
- **Learning Rate**: Increasing η to 0.01 boosts performance for FedAvg (+5.75%) but may upset Krum.
- **FedChallenger** retains its robust characteristics across different configurations, though $B = 10$ surprisingly outperforms at $\eta = 0.001$.

The findings of this study demonstrate the importance of robust aggregation techniques within FL, particularly in adversarial environments. *FedChallenger* proved to be the most effective method for sustaining high performance despite considerable adversarial influence. Consequently, it represents a compelling candidate for real-world deployments where device data poisoning presents a

significant challenge. While the default parameters ($B = 20$, $\eta = 0.001$) achieve a balance of stability and performance, fine-tuning η holds the potential for even greater mitigation of poisoning effects.

4.5 Chapter Summary

Current defence mechanisms against model and data poisoning attacks demonstrate insufficient protection against malicious sample propagation during the aggregation process. To address these limitations, this work introduces *FedChallenger*, a zero-trust challenge-response framework that leverages device behavioural analysis for detecting compromised data or model parameters. The system incorporates a dual-mode adaptive defence strategy, combining cosine similarity-based consensus boosting during regular operation with *MAD*-based filtering when attacks are detected. Comprehensive evaluation across MNIST, EMNIST, FMNIST, and CIFAR-10 datasets demonstrates *FedChallenger*'s consistent superiority, achieving 8.2%, 47.8%, 88.4%, and 7.4% higher accuracy than Trimmed-Mean, Krum, FedAvg, and DUEL, respectively, under 30% Label Flipping attacks on FMNIST. The technique also shows significant computational advantages, converging 60%, 52%, 31%, and 27% faster than these baseline methods in 20% Label Flipping scenarios. For more severe 40% model poisoning attacks on CIFAR-10, *FedChallenger* maintains substantial performance leads, outperforming comparative approaches by factors ranging from 1.24 to 6.7 times. When evaluated against contemporary techniques, *FedChallenger* demonstrates 36%, 47%, and 54% faster convergence than Stake, Shap, and Cluster, respectively, on the CIFAR-10 dataset. These results validate *FedChallenger* as a robust, dataset-independent solution that effectively addresses the limitations of existing defence mechanisms through its multi-layered architecture, combining challenge-response verification with enhanced consensus-based aggregation, proving particularly effective at neutralising diverse poisoning threats while maintaining model performance across all tested scenarios.

Chapter 5

SignDefence: Byzantine-Robust Federated Learning with Sign Direction and Leaky ReLU

In this chapter, we introduce *SignDefence*, a sign direction and LeakyReLU-based aggregation technique, which considers the direction of the gradients and overcomes the performance issues related to the dying ReLU problem. Moreover, the proposed *SignDefence* computes Jaccard Similarity over binary encoded model weights and remains robust across sparse data. The following sections details the functional component of *SignDefence* and its significance with experimental validity.

5.1 SignDefence and Its Contributions

While FL offers numerous advantages, malicious actors may target its weaknesses to execute either model poisoning or data poisoning attacks [41]. To protect these systems, robust and efficient aggregation mechanisms are essential. Numerous studies have already proposed to lessen the impact of such attacks through various aggregation methods. Notable examples include Krum [15], Trimmed-Mean [143], and FedAvg [79], which rely on robust aggregation techniques. Other approaches, like Flod [32], are based on sign direction, while Flame [86] uses clustering to detect

and remove corrupted samples. However, each of these methods comes with its own set of challenges. FedAvg, the foundational aggregation technique rooted in the Federated Averaging (FA) algorithm [79], remains susceptible to attacker modified gradients. Krum aims to find suboptimal gradients near the median but assumes that client data is IID [34]. This assumption often doesn't hold true, which can affect Krum's performance. While Trimmed-Mean is considered more robust, it discards a certain percentage of gradients from both ends after sorting the data. This trimming of data points leads to information loss, and accurately estimating the optimal cutoff proportion is difficult, often negatively impacts its effectiveness. To address precise trimming and better identify poisoned samples, the Flame strategy employs HDBSCAN-based clustering with cosine similarity as a distance measure [86]. Yet, Flame's performance is sensitive to hyperparameters like noise levels and clustering thresholds. Flod, recognized as an efficient sign direction technique for federated aggregation, also faces issues due to hyperparameter sensitivity and the dying ReLU problem [146], especially when considering τ-clipping with Hamming distance and the Rectified Linear Unit (ReLU) activation [32].

Most current state-of-the-art aggregation strategies struggle with either incorrectly removing legitimate samples or, conversely, including poisoned samples as valid updates. Furthermore, many of these techniques are highly sensitive to increased noise and often demand careful hyperparameter tuning. Although Flod successfully addresses most data clipping problems using sign-magnitude and Hamming distance-based τ-clipping, it still falters with increased data noise. Consequently, poisoning attacks significantly degrade Flod's performance, and it additionally contends with the dying ReLU problem. This highlights a clear need for an updated strategy that can aggregate data efficiently while considering these persistent challenges.

In response to these issues, this chapter introduces a new variant of the Flod technique, named *SignDefence*. This proposed method specifically addresses noisy gradients and Flod's inherent dying ReLU problem. Our key contributions include:

- Design the *SignDefence* technique, which uses the sign magnitude of gradients and LeakyReLU with Jaccard Similarity for weight estimation to effectively mitigate poisoning attacks.
- Present an FL architecture that utilizes *SignDefence* aggregation strategies.

- Propose the *SignDefence* algorithm to train the model with no or minimal impact from poisoning samples.

- Evaluate the performance of the proposed *SignDefence* technique using the HAR dataset [41] against state-of-the-art Krum, Trimmed-Mean, FedAvg, Flame, and Flod techniques. Furthermore, these evaluations were rigorously conducted under scenarios including no-attack, attack on Trimmed-Mean, attack on Krum, Min-Max attack, Min-Sum attack, and Label Flipping attack.

5.2 Comprehensive System Architecture

This section outlines the architectural framework and operational methodology of the proposed *SignDefence* aggregation approach.

The *SignDefence* technique uses a structured modular architecture as shown in Fig. 5.1. This architecture enables collaborative model training between multiple participating client devices $C \in i$ and a central federated server (FS) component. The federated server includes several important operational modules that work together: The Device Selector (DS) module implements client selection mechanisms using either random sampling techniques or more advanced trust-based and performance-based selection strategies as documented in [39]. The Federated Training Controller (FTC) serves as the central coordination hub that manages the complete federated training lifecycle. It maintains constant communication with all other FS modules while orchestrating client device interactions and issuing training directives throughout the learning process. The Aggregation module represents the core defensive component of the system. It consists of multiple specialized computational submodules that collectively provide robust protection against various threats. These include the Weight Accumulator (WA) that receives and distributes model parameters, the Binary Encoder (BE) that performs critical encoding operations, the Median Function (MF) that calculates central tendency metrics, the Sign Computation Function (SCF) that processes weight signatures, the Jaccard Similarity (JS) module that analyzes model update relationships, the LeakyReLU Module (LM) that implements advanced activation functions, and finally the Weight Estimator (WE) that computes the final aggregated updates. The WA submodule initially receives the model weights

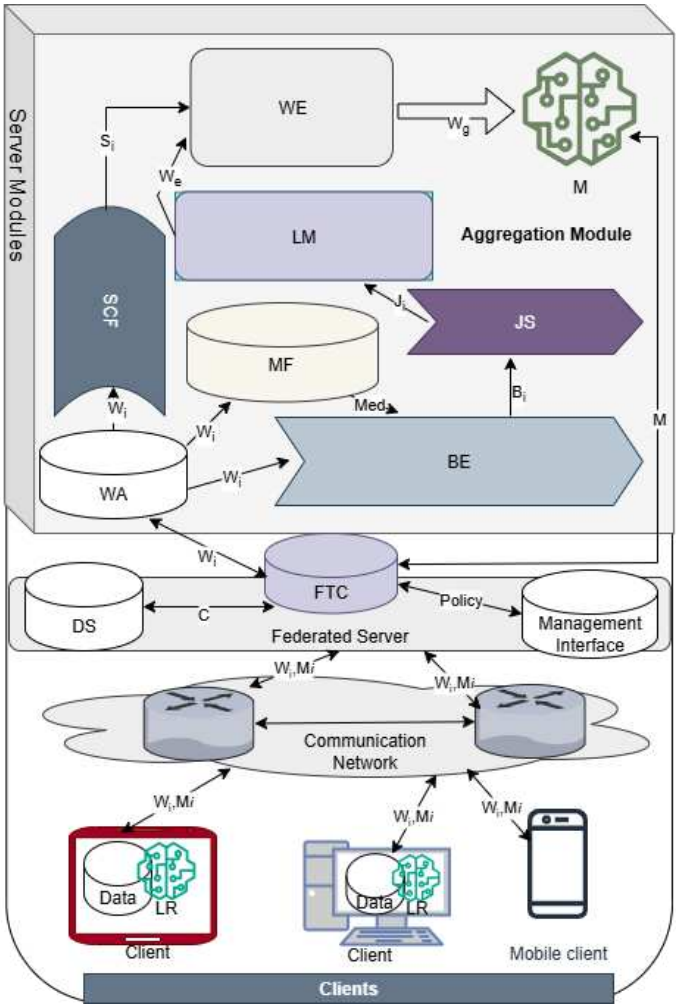


Figure 5.1: Complete Architectural Schematic of SignDefence Framework

W_i from the FTC and distributes them to three parallel processing pipelines: the SCF submodule processes W_i to derive the mathematical signs S_i of the weights according to Eq. (10), while the MF submodule calculates the median value Med of the weights through Eq. (11), and the BE submodule generates binary encoded representations B_i of the weights using Eq. (12).

$$S_i = \{\text{Sign}(W_k)\}_{k=0}^{i} \tag{10}$$

$$Med = Median(W_i) \tag{11}$$

$$B_i = \{\mathbb{I}(-\text{Med} < W_k < \text{Med})\}_{k=0}^{i} \tag{12}$$

The JS submodule processes the binary encoded weights B_i to compute pairwise similarity metrics J_i following the mathematical procedure outlined in Eq. (13). This similarity measure was selected for its robustness against noise contamination and its ability to effectively handle input sequences of varying lengths. The LM submodule then generates preliminary weight estimates J_k values against a predefined security threshold th through the application of a LeakyReLU activation function as defined in Eq. (14). This nonlinear transformation was chosen over standard ReLU implementations due to its superior ability to prevent dying ReLU problem that frequently occur in deep learning systems.

$$J_i = \{J(B_k, B_i)\}_{k=0}^{i} \tag{13}$$

$$W_e = \{\text{LeakyReLU}(th - J_k)\}_{k=0}^{i} \tag{14}$$

The WE submodule performs the final aggregation step by combining the estimated weights W_e with their corresponding signs S_i to produce the ultimate global model updates W_g through the weighted aggregation formula specified in Eq. (15). These computed updates are then applied to the current global model M, with performance evaluation conducted after each training round r. The FTC initiates subsequent training rounds by coordinating with the DS module to select appropriate client devices for participation, while system administrators can configure various training hyperparameters and establish operational rules through the dedicated Management Interface. On the client side, each participating device independently executes local Linear Regression (LR) computations using their respective private datasets while incorporating the most recent global model M_r and associated weights W_r received during each communication round r.

$$W_g = \frac{\sum_{k=0}^{i} W_{e,k} \cdot S_k}{\sum_{k=0}^{i} W_{e,k}} \tag{15}$$

Algorithm 4 provides the complete formal specification of the *SignDefence* aggregation methodology in pseudocode format. The procedure begins by initializing the global model M_0 with its corresponding parameters W_0 and selecting appropriate participant devices through a parallelized selection process. During each training round r within the total number of rounds R, every participating client device independently computes its local model updates W_i by executing Linear

Regression (LR) using the current global model parameters. The algorithm then processes these collected updates through multiple sequential defensive operations: First computing the mathematical signs of all weights, then determining the median value of the weight distribution, followed by generating binary encoded representations of the model parameters, and subsequently analyzing pairwise similarities between client updates.

Algorithm 4 SignDefence Aggregation Procedure

Input:

Initial global model M_0 with parameters W_0

Security threshold th

Total training rounds R

Client selection pool $\mathcal{C}$

Output: Robust global model M

$M \leftarrow M_0$ *(Global model initialization)*

$W \leftarrow W_0$ *(Weight initialization)*

for each round $r \in R$ **do** *(r in total rounds R)*

  $\mathcal{C}_r \leftarrow \text{SELECTCLIENTS}(\mathcal{C})$ *(Parallel client selection)*

  for each client $i \in \mathcal{C}_r$ **do**

    $W_i \leftarrow \text{LOCALLR}(M, W)$ *(Client i's local update)*

  end for

  $S_i \leftarrow \{\text{sign}(W_{i,k})\}_{k=1}^{n}$ *(Signs for client i's weights)*

  $\text{Med} \leftarrow \text{median}(\{W_i\}_{i \in \mathcal{C}_r})$ *(Median weight)*

  $B_i \leftarrow \{\mathbb{I}(-\text{Med} < W_{i,k} < \text{Med})\}_{k=1}^{n}$ *(Binary encoding)*

  $J_i \leftarrow \{J(B_k, B_i)\}_{k \in \mathcal{C}_r}$ *(Pairwise similarities)*

  $W_e \leftarrow \{\text{LeakyReLU}(th - J_{i,k})\}_{k=1}^{n}$ *(Estimated weights)*

  $W_{g,r} \leftarrow \frac{\sum_{k=1}^{n} W_{e,k} \odot S_k}{\sum_{k=1}^{n} W_{e,k}}$ *(Aggregated weights)*

  $M \leftarrow M \leftarrow W_{g,r}$ *(Model update)*

end for

return M

The LeakyReLU activation function is then applied to estimate robust weights while mitigating potential gradient issues, ultimately leading to the computation of final aggregated weights $W_{g,r}$ through a weighted combination of the estimated weights and their corresponding signs. These aggregated weights are then incorporated into the global model M_r for the current round. This

iterative process continues until global model reaches the stable peak performance, at which point the final robust global model M is produced as the output of the complete *SignDefence* procedure.

5.3 Evaluation Results

The complete simulation was carried out using PyTorch and SAFEFL [41] on a Macbook M1 Pro chip, featuring an 8-core CPU, 14-core GPU, 16GB of RAM, and a 512GB SSD. The FL configuration included 30 clients and a single server. The chosen HAR [41] dataset encompasses data from six activities performed by thirty human subjects, with data distributed in a non-IID manner among the participating clients. For training, a batch size of 64 was selected, and the learning rate was set to 0.005. The evaluations employed a Linear Regression (LR) classifier, comparing the performance of *SignDefence* against Trimmed-Mean [143], Krum [15], Flod [32], FedAvg [79], and Flame [86] techniques. These comparisons were conducted across several scenarios: no-attack, attack on Krum, attack on Trimmed-Mean, Label Flipping attack, Min-Max attack, and Min-Sum attack.

In the Krum attack scenario, the local model parameter (W') is intentionally crafted from a compromised model parameter (C') to maximize its deviation from the true global parameter (W). For the Trimmed-Mean attack, the adversary manipulates the local model parameter W' using either the minimum or maximum of benign local model parameters, depending on the inverse direction of the global model parameters. A Label Flipping attack involves the attacker substituting one class label for another to degrade the model's performance. In a Min-Max attack, a malicious weight (δW_k) is chosen such that its maximum distance from other weights W_i is less than the maximum distance between any two benign weights. Conversely, during a Min-Sum attack, the attacker identifies a δW_k whose sum of distances from other weights is less than the sum of distances between other benign weights. These evaluations assumed that 20% of the client's total model parameters and data were compromised. Each experiment ran for 2000 fixed training rounds, with models assessed after every training round.

To establish the robustness of the proposed *SignDefence* technique, its performance was benchmarked against the state-of-the-art Trimmed-Mean [143], Krum [15], Flod [32], FedAvg [79], and

Flame [86] techniques. The evaluation metrics included training accuracy and average F1 Score. Accuracy quantifies the overall proportion of correct predictions, while the F1 score offers a balanced measure of model performance by combining precision and recall. In each communication round, the global model (M) was updated, and this updated M was then tested with the test dataset to measure its accuracy. The average F1 score was computed at the conclusion of all communication rounds. These evaluation measures were repeated for all techniques and attack combinations. The subsequent subsections present the detailed evaluation results and discussions.

5.3.1 No-Attack Scenario

In the no-attack scenario, no data or model parameters were intentionally altered. Fig. 5.2 illustrates the accuracy of *SignDefence* in comparison to the state-of-the-art techniques. The evaluation results indicate that the proposed *SignDefence* is 10% more accurate than Krum and approximately 7% more accurate than Trimmed-Mean, Flame, and FedAvg under no-attack conditions. Krum's sub-optimal selection of model parameters closest to the median contributes to its performance drop. In contrast, Trimmed-Mean's accuracy suffers because it discards important data trends through its trimming process. FedAvg's lower accuracy can be linked to the data distribution and the drift of local client models from the global model. Meanwhile, Flame's sensitivity to hyperparameters and local evaluation metrics may lead to degraded accuracy.

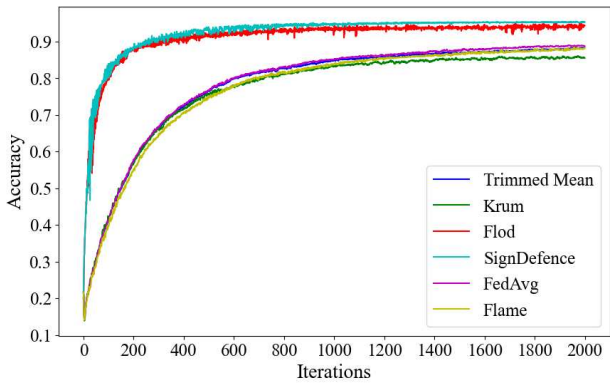


Figure 5.2: Accuracy Vs Iterations Under No-Attack Scenario.

However, *SignDefence* does not remove any information and relies on the signed direction of

the model parameters. Furthermore, it estimates model updates by considering the signed magnitude and Jaccard Similarity of binary encoded gradients, and it employs LeakyReLU for weight estimation. *SignDefence*, being a variant of Flod, exhibits accuracy similar to the Flod technique. Nevertheless, it shows a 1% improvement due to its incorporation of median-based binary encoding, Jaccard Similarity, and LeakyReLU.

5.3.2 Attack On Krum

The attack on Krum was executed with 20% malicious clients, following the attack procedure detailed in [34]. Fig. 5.3 presents the evaluation results of the proposed *SignDefence* compared to existing techniques. Since this attack is specifically designed for the Krum technique, Krum experiences a substantial accuracy drop. Compared to the proposed *SignDefence*, Krum is 56% less accurate because it selects sub-optimal model parameters based on median difference. Conversely, *SignDefence* demonstrates consistent performance under the attack on Krum scenario as it utilizes Jaccard Similarity measures between binary encoded data. It estimates model parameters using LeakyReLU and incorporates the sign magnitude of weights along with the estimated ones. For similar reasons, it shows approximately a 10% accuracy improvement over Trimmed-Mean and an 8% improvement over both FedAvg and Flame techniques. Although the final accuracy of Flod and *SignDefence* do not differ significantly, Flod appears more susceptible to outliers.

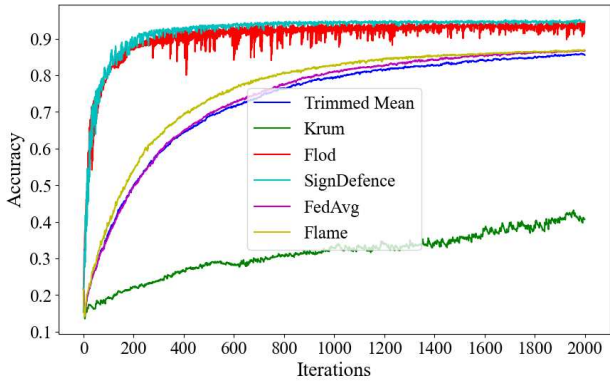


Figure 5.3: Accuracy Vs Iterations Under Attack on Krum.

5.3.3 Attack on Trimmed-Mean

The attack on Trimmed-Mean was performed according to the process outlined in [34], involving 20% malicious clients. Fig. 5.4 displays the results of the attack on Trimmed-Mean for *SignDefence* and its counterparts. Flod is the poorest performer in this evaluation, losing 15% accuracy compared to *SignDefence*. Similarly, FedAvg, Krum, Trimmed-Mean, and Flame are 12%, 5%, 8%, and 4% less accurate, respectively, than the proposed *SignDefence* technique. Flod's reliance on Hamming distance-based τ-clipping and boolean encoding strategy did not adapt well to changes in data. However, *SignDefence* consistently maintains its accuracy curve and shows less sensitivity to outliers due to its use of LeakyReLU and sign direction for weight estimation.

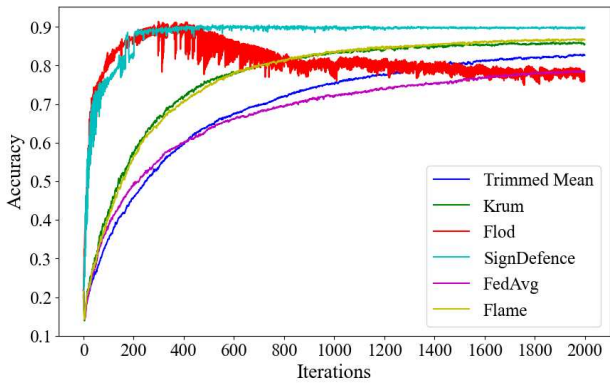


Figure 5.4: Accuracy Vs Iterations Under Attack on Trimmed-Mean.

5.3.4 Label Flipping Attack

In this experiment, 20% of clients had their labels flipped to another class. Fig. 5.5 presents the results of the Label Flipping attack for Trimmed-Mean, Krum, Flod, *SignDefence*, FedAvg, and Flame strategies.

The experimental results indicate that the proposed technique is nearly 9% more accurate than both Krum and FedAvg, attributable to their sensitivity to class distribution. Additionally, Trimmed-Mean and Flame are approximately 6% less accurate than *SignDefence*. Being a variant of Flod, *SignDefence* still performs better under a Label Flipping attack, with an accuracy gain of almost 1% compared to Flod.

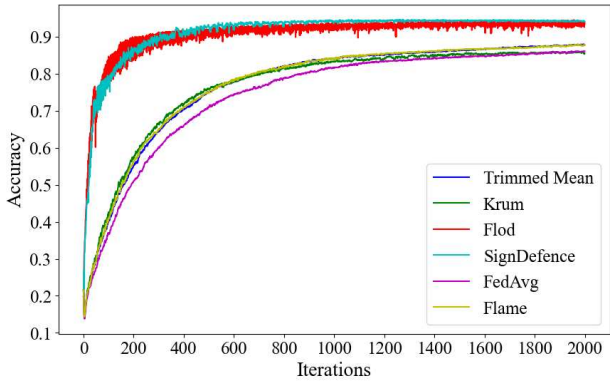


Figure 5.5: Accuracy Vs Iterations Under Label Flipping Attack.

Table 5.1: F1 Score for State-of-the-art Techniques Under Different Attacks

Attack / Technique	No	Trimmed-Mean	Min-Sum	Krum	Label Flipping	Min-Max
Trimmed-Mean	0.78	0.69	0.73	0.72	0.77	0.71
Krum	0.76	0.76	0.32	0.28	0.76	0.26
Flod	0.91	0.80	0.93	0.90	0.91	0.84
SignDefence	0.92	0.88	0.94	0.92	0.91	0.88
FedAvg	0.78	0.65	0.74	0.73	0.74	0.73
Flame	0.77	0.76	0.75	0.76	0.77	0.75

5.3.5 Min-Max Attack

In this evaluation, the Min-Max attack was simulated using the technique described in [100]. Fig. 5.6 presents the evaluation results of the Min-Max attack on Trimmed-Mean, Krum, Flod, *SignDefence*, FedAvg, and Flame techniques. The evaluation results suggest that Krum's accuracy drops by nearly 67% compared to the *SignDefence* strategy, primarily due to its sensitivity to outliers and sub-optimal selection of model parameters. Conversely, *SignDefence* estimates weights based on the sign magnitude of the weights and LeakyReLU computation, incorporating Jaccard Similarity, which operates on binarized weights. This allows *SignDefence* to follow the direction of weight changes and maintain consistent performance. For similar reasons, Trimmed-Mean, FedAvg, and Flame techniques are 8%, 5%, and 4% less accurate than *SignDefence*, respectively. However, Flod predominantly struggles due to its reliance on ReLU and Hamming distance-based clipping strategy. Consequently, it shows an over 6% accuracy drop compared to the proposed *SignDefence* technique.

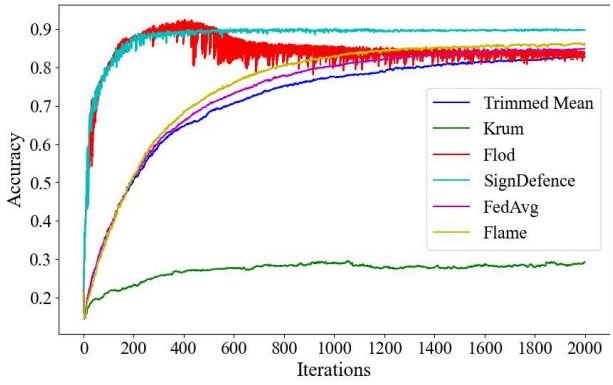


Figure 5.6: Accuracy Vs Iterations Under Min-Max Attack.

5.3.6 Min-Sum Attack

Fig. 5.7 presents the results of the Min-Sum attack on the Trimmed-Mean, Krum, Flod, *SignDefence*, FedAvg, and Flame techniques. Here, the Min-Sum attack was engineered according to the strategies presented in [100]. Since the Min-Sum attack aims to minimize the sum of distances among all weights, almost all existing techniques, with the exception of Krum, exhibit consistent performance comparable to the no-attack scenario. Krum, however, shows nearly 65% degraded accuracy compared to *SignDefence* because it relies on selecting suboptimal model parameters closest to the median. Trimmed-Mean, FedAvg, and Flame are nearly 11% less accurate than *SignDefence*, which can be attributed to the proposed technique's sign-magnitude and LeakyReLU-based implementation. In this instance, Flod again demonstrates accuracy similar to the proposed *SignDefence*, with the proposed method being 1.3% more accurate than Flod due to its incorporation of LeakyReLU and sign magnitude-based model parameter estimation.

5.3.7 F1-Score Analysis

Finally, this subsection presents the average F1 score for *SignDefence*, Krum, Trimmed-Mean, Flod, Flame, and FedAvg techniques under no-attack and various attack scenarios. Table 5.1 indicates that the proposed *SignDefence* is robust irrespective of the different attack types and consistently offers a better F1 score than its counterparts.

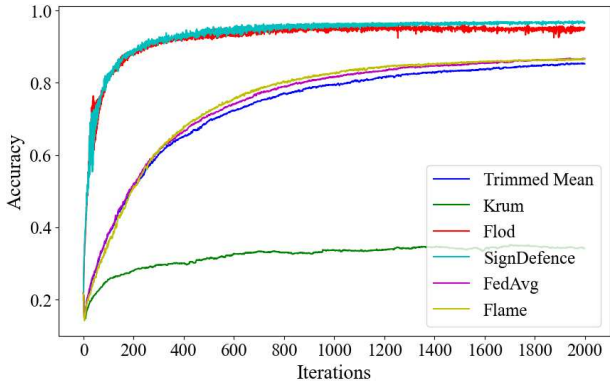


Figure 5.7: Accuracy Vs Iterations Under Min-Sum Attack.

Since *SignDefence* relies on the sign magnitude that defines the direction of the gradients, it performs effectively regardless of compromised samples. Furthermore, its weight estimation, which uses LeakyReLU and incorporates Jaccard Similarity, resolves the dying ReLU problem and performs better on highly skewed data distributions. Techniques other than Flod generally show much lower F1 values, which is also reflected in their accuracy results. *SignDefence*, as a variant of Flod, follows a similar performance pattern, though some attack types cause Flod to exhibit significantly degraded performance because of its reliance on ReLU and Hamming distance-based clipping.

5.4 Chapter Summary

Poisoning attacks targeting Big Data-dependent AI systems pose significant threats and can undermine their practical benefits, resulting in substantial losses. Current defence strategies primarily depend on robust aggregation techniques or anomaly clustering methods. However, these approaches often fail to effectively eliminate the influence of malicious samples and sometimes lead to the unintended removal of legitimate samples. This research introduces *SignDefence*, a novel Flod technique variant that employs the Sign direction strategy combined with LeakyReLU and Jaccard Similarity for model weight estimation. Evaluation results on the HAR dataset under various attack scenarios demonstrate that the proposed technique maintains consistency and surpasses all existing state-of-the-art defence mechanisms.

Furthermore, *SignDefence* demonstrates robustness and consistently achieves superior F1 scores

across all attack and non-attack scenarios. This improvement stems from its consideration of gradient sign direction and its solution to the dying ReLU problem.

Chapter 6

Fed-Reputed: Reputation-Aware Client Selection in Hierarchical Federated Learning

This chapter introduces *Fed-Reputed*, a reputation-aware client selection framework tailored for HFL. Unlike existing reputation-based approaches that suffer from herding effects, cold-start limitations, and imbalanced classification datasets, *Fed-Reputed* integrates a modified Bellman equation within a Deep Q-Learning framework. This formulation guides client selection using an Imbalanced Classification Markov Decision Process (ICMDP), leveraging both device capability and historical behavior.

The following sections highlights the working principle and significance of the proposed *Fed-Reputed* technique.

6.1 Motivations and Contributions

As discussed earlier, centralized FL systems [14] utilize a coordinating server that manages distributed training across client devices, keeping data close to its source. The global model undergoes continuous improvement from aggregated updates gathered from clients after each training cycle. This structure ensures that sensitive information stays on the originating device, reducing the risk of

privacy breaches. However, such centralized models face challenges in scaling and adapting, mainly when used across diverse networks with mobile health devices [2]. As a result, HFL has emerged as a practical solution within the FL research community. HFL designs add intermediate edge servers positioned near client groups, which assist the central server [129]. These edge nodes handle local data aggregation before sending information to the central server, creating a multi-level learning system. This design is especially beneficial for resource-constrained consumer electronics used in healthcare environments, where constant communication with a central server may not be feasible or could drain energy. Nevertheless, real-world FL applications face notable operational challenges. Client devices vary significantly in their computing power, network connectivity, and data quality characteristics [39]. Straggler nodes (devices that struggle with poor connectivity or processing capability takes indefinite time to respond) and malicious users (potentially compromised by attackers) can significantly hinder system performance by delaying convergence or launching model poisoning attacks [98, 153]. In privacy-sensitive biomedical settings, such behaviour can lead to serious ethical, security, and health consequences. Since it is not feasible to achieve an environment entirely free of malicious or straggler nodes, it is crucial to develop effective client selection methods to reduce their impact on model training while maintaining accuracy and speeding up convergence.

Current FL methods mainly use random client selection (RS), performance-based strategies, cluster-based methods, weighted tactics, and reputation-based choices [79, 94, 77]. Randomly selecting clients during each training round does not fully leverage local updates from diverse participants, leading to lower model accuracy, longer convergence times, and potential fairness issues [39]. Moreover, RS often includes malicious and straggler clients in the selection pool, causing significant drops in model performance [124]. This problem is especially severe in semi-supervised biomedical contexts, where limited labelled data necessitates maintaining model accuracy. Performance-driven strategies [39] assess the training effectiveness of clients to make informed selection decisions. However, metrics often fail to reflect the full characteristics of clients accurately. Cluster-based methods [37] aim to group similar clients and pick representatives from different distributions. Nonetheless, there is still a risk of including malicious and straggler participants, as resource-limited or misbehaving clients might be selected during training. Weighted selection methods [84] prioritize capable devices but may lead to biased model updates and unfair selections. Additionally, these

strategies do not eliminate the risk of including malicious and straggler clients. Thus, reputation-based client selection has gained traction in FL research [77]. These methods assign reputation scores to clients based on their past behaviour, excluding those below certain thresholds from selection. The threshold values can change dynamically, and the reputation scores are calculated using deep deterministic policy gradients [77]. While more robust, many of these approaches encounter cold-start issues, as they need a sufficient history of interactions to make informed decisions [125]. The Stochastic Client Selection (SCS) method [113] tackles cold-start challenges by considering uncertainties. It also accounts for client hiring costs and includes participants with varying reputation levels during training rounds. However, it lacks support for HFL architecture and might unintentionally select malicious clients in initial rounds. On the other hand, Multi-Agent Deep Deterministic Policy Gradient (MADDPG) methods [77] effectively manage HFL but still face cold-start challenges and require subsequent interaction data for accuracy. Unfortunately, malicious clients might compromise the models before these strategies reach their optimal state. Furthermore, the distribution of malicious or straggler nodes compared to benign clients is not uniform across communication rounds, resulting in imbalanced datasets. Therefore, there is a strong need for reliable client selection frameworks for HFL that can use available device data, effectively address cold-start issues, and identify misbehaving clients without compromising model security and privacy. Additionally, these frameworks should function smoothly within hierarchical architectures and support FL setups by ensuring dependable client participation while taking both current and historical data into account and addressing dataset imbalance.

This research fills these gaps by presenting *Fed-Reputed*, a DQN-based reputation-driven client selection framework created explicitly for HFL in consumer biomedical devices. It incorporates a Deep Reinforcement Learning (DRL) strategy that utilizes DQN, along with an ICMDP classifier [65], for the unbiased selection of edge servers and consumer clients, addressing the growing privacy concerns surrounding consumer electronics. Specifically, ICMDP leverages device capability information to tackle cold-start challenges. The key contributions of this research work are outlined below:

- Development of the *Fed-Reputed* framework, drawing inspiration from cutting-edge DQN methodologies and ICMDP techniques to facilitate optimal consumer client selection across

both initialization and standard training phases.

- Design of a HFL architecture that harnesses ICMDP capabilities to effectively identify and impose penalties on malicious participants and straggler devices.
- Formulation of an enhanced Bellman equation that integrates reputation scoring mechanisms derived from historical Q-values, straggler identification, malicious behaviour detection, training performance metrics, and computational resource utilization.
- Comprehensive performance assessment of *Fed-Reputed* using MNIST and FMNIST benchmark datasets, with comparative analysis against SCS [113], MADDPG [77], and RS [79] methodologies across diverse straggler and malicious participant configurations.

6.2 Fed-Reputed Architecture and Algorithms

This section outlines the architectural framework and operational methodology of the proposed *Fed-Reputed* approach. The system architecture, depicted in Fig. 6.1, comprises three fundamental components: a federated server S, multiple edge servers E, and participating consumer clients I. The central server S implements management directives through its dedicated sub-module while overseeing the model training workflow.

The server infrastructure incorporates three core sub-modules: the Federated Controller (FC), Aggregator (AG), and device selection (DS) mechanism. The DS component integrates a DRL-based selection mechanism that identifies suitable edge servers E, which are subsequently maintained in a device pool for round-robin allocation [5] during global communication rounds. The FC manages the federated training process by coordinating with selected edge servers and consolidating their model updates (W_e) via the AG sub-module. These aggregated updates ($\mu W'$) are applied to the baseline model (M_0), with the enhanced model's performance determining subsequent training iterations until convergence criteria are satisfied.

Each edge server $e \in E$ implements four principal components: an agent program (AP), Edge Aggregator (EA), Edge Training Controller (ETC), and a DRL module with associated device pool.

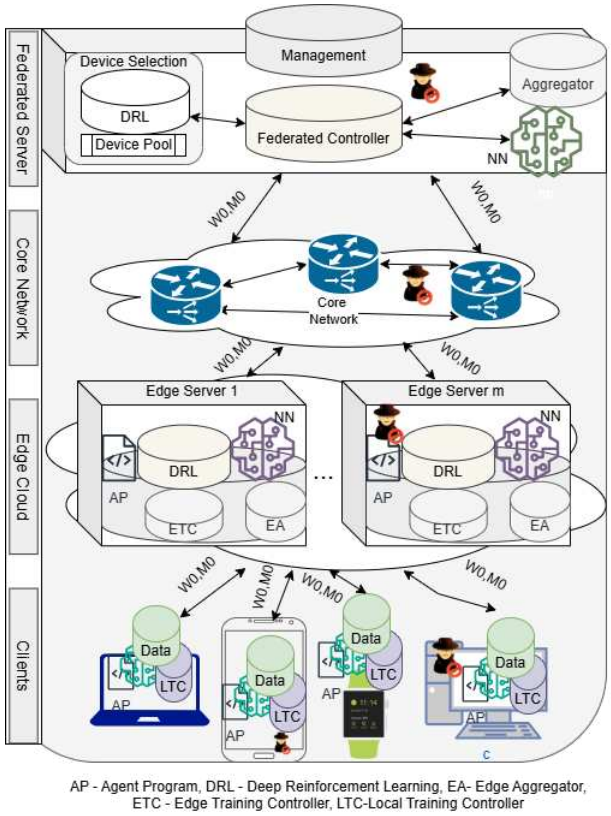


Figure 6.1: Architectural Overview of Fed-Reputed Framework

The *AP* facilitates background operations including metadata exchange, cryptographic key management, and participation request processing. The *ETC* orchestrates local training operations and performs intermediate aggregation with *EA* following each edge round $er \in ER$. The DRL module processes client participation requests $ID \in I$ through a DQN algorithm with ICMDP classification [65], evaluating devices based on behavioural metrics.

The ICMDP classifier utilizes a modified Bellman equation incorporating reputation scoring through Eq. (16), where RS_{t+1} derives from previous state parameters including Q_t, RS_t, $RU_{I,t}$, GA_t, SS_t, and MS_t, aggregated over intervals $l = 1$ to t. The resource utilization metric RU_t encompasses computational resources including processing capacity, memory allocation, and energy consumption, with a scaling factor ψ normalizing reputation scores.

$$RS_{t+1} = \psi Q_t RS_t \frac{(RU(I,t) + \theta GA_t + \alpha SS_t + \beta MS_t)}{\sum_{l=1}^{t}(GA_l + SS_l + MS_l)} \tag{16}$$

The Q-value computation under policy π for state-action pair (s,a) follows Eq. (17), integrating RS_{t+1} with expected rewards and discounted future values. This Q-value feeds into the ICMDP loss function, ultimately yielding converged Q-network parameters that, when combined with a softmax layer [65], form the device selection classifier. Compliant devices enter the round-robin scheduling pool [5] for subsequent training rounds.

$$Q^{\pi}(s,a) = RS_{t+1} * E_{\pi}[r_t + \gamma Q^{\pi}(s_{t+1}, a_{t+1}) | s_t = s, a_t = a] \tag{17}$$

Algorithm 5 formalizes the device selection process, accepting inputs ID, T, MD^{π}, and weighting factors θ, α, β. The procedure collects device metrics including RU, GA, SS, and MS during each training round $t \in T$, computing $RS_{I,t+1}$ via Eq. (16) and subsequent Q-values through Eq. (17) for ICMDP processing.

Algorithm 5 Device Selection Algorithm

Input:
Training Rounds, T,
Accuracy, Malicious, and Straggler weight factors θ, α, and β, respectively
Management Defined Selection Policy, MD^{π},
Q-value at state t, Q_t

Output:
Selected Devices (SD)

$ID = getInterestedDevices()$
for $t \in T$ **do**
  for $I \in ID$ **do**
    $RU_{I,t} = getDeviceResourceUsage(I,t)$
    $GA_{I,t} = getGlobalAccuracyContrib(I,t,M_t)$
    $SS_{I,t} = getStragglerStatus(I,t,SS_{I,t-1})$
    $MS_{I,t} = getMaliciousStatus(I,t,MS_{I,t-1})$
    $RS_{I,t+1} = \psi Q_t RS_t \frac{(RU(I,t)+\theta GA_t+\alpha SS_t+\beta MS_t)}{\sum_{I=1}(GA_I+SS_I+MS_I)}$
    RUN $ICMDP$ with Q-value computed using $RS_{I,t+1}$ by Eq. (17)
    $DC_t = getSoftmaxAddedQNetwork(t)$
    if $DC_t(I) \in MD^{\pi}$ **then**
      $SD_t = SD_t \cup I$
    else
      $SD_t = SD_t \cap I$
    end if
  end for
end for
Return SD

The ICMDP algorithm maintains state memory for DQN convergence, with the resulting Q-network and softmax layer forming the Device Classifier (DC) that governs device pool membership. Consumer clients implement AP, LTC, and local datasets, with LTC employing mini-batch

SGD [79] for local model training.

The training protocol coordinates operations across S, E, and C components, initiating with parameters W_0 and M_0. Edge servers execute Algorithm 6, incorporating parallel device selection that collects DMI during each $er \in ER$. Client devices C_{er} are selected via round-robin scheduling, with accepted models satisfying SD^{π} criteria through importance metric IM evaluation.

Algorithm 6 HFL Training at Edge Server, E

Input:
Initial Global Model, M_0
Initial Model Parameters, W_0
Selected Device Policy, SD^{π}
Output:
Global Model, M

RUN Parallel $SD_C = getSelectedDevices(\text{DMI})$
for $er \in ER$ **do**
  $M_{er-1} = M_0$
  $W_{er-1} = W_0$
  $C_{er} = sample(SD_C)$
  for $c \in C_{er}$ **do**
    $DMI_c = getDevicePMI(er)$
    $M_{c,er} = train(c, M_{er-1}, W_{er-1})$
    $DMI = DMI \cup DMI_c$
    if $IM(c) \in SD^{\pi}$ **then**
      Accept $M_{c,er}$
    end if
  end for
  $M_{er} = M_{er-1} \cup \frac{\sum_{j=1}^{count(E_{er})} M_{c,er}}{count(E_{er})}$
end for
Edge Model After All Edge Round, $M = M_{er}$
return M

Following edge round completion, aggregated parameters form M_{er}, with comprehensive edge models transmitted to S for final aggregation using Algorithm 6. The server iteratively evaluates model performance after each communication round until it achieves the desired performance.

6.2.1 System Model and Definitions

We rigorously characterize *Fed-Reputed*'s framework through the following core mathematical constructs:

Definition 1 (FL Objective)**.** *The global model* $\mathbf{w}$ *is trained by minimizing:*

$$\min_{\mathbf{w}} F(\mathbf{w}) = \mathbb{E}_{i \sim \mathcal{P}}[F_i(\mathbf{w})] \tag{18}$$

where $F_i(\mathbf{w}) = \mathcal{L}(\mathbf{w}; \mathcal{D}_i)$ represents the local loss function calculated over client i's local dataset $\mathcal{D}_i$, with $\mathcal{P}$ denoting the client's data distribution. This formulation ensures non-IID data distribution across clients while safeguarding privacy by decentralized data.

Definition 2 (Reputation Update Strategy)**.** *The reputation score for client i, $RS_t(i)$ at communication round t is obtained by:*

$$RS_t(i) = \psi Q_{t-1}(i) RS_{t-1}(i) \frac{RU_t(i) + \theta GA_t(i) + \alpha SS_t(i) + \beta MS_t(i)}{\sum_{\tau=1}^{t} (GA_\tau(i) + SS_\tau(i) + MS_\tau(i))} \tag{19}$$

where:

- *$RU_t(i)$: Resource utilization metric consisting of CPU, memory, and bandwidth consumption*
- *$GA_t(i)$: Global accuracy contribution (measured by validation performance)*
- *$SS_t(i)$: Straggler detection indicator (binary flag for delayed responses)*
- *$MS_t(i)$: Malicious behavior flag (gradient deviation detection)*
- *ψ: Learning adaptation parameter to control reputation score*
- *θ, α, β: Accuracy, Malicious, and Straggler weight factors*
- *$Q_{t-1}(i)$: Q-value from previous round*

This equation dynamically adapts client trust level based on multiple performance metrics to balance computational efficacy, model accuracy, and system robustness.

6.2.2 Straggler Mitigation Strategy

Fed-Reputed detects and overcomes straggler's impact through its reputation-oriented client selection strategy:

Theorem 3 (Straggler Detection Bound)**.** *Consider any client i that exhibits straggler behavior, defined by a delay Δt_i exceeding a predefined threshold τ_{th}. The likelihood of selecting this client at time t, denoted by $p_t(i)$, diminishes exponentially as follows:*

$$p_t(i) \leq \exp\left(-\frac{\alpha^2 t}{2\sigma_{SS}^2}\right) \tag{20}$$

Here, σ_{SS}^2 represents an upper bound on the variance observed in the patterns of straggler behavior, and α is the weight assigned to the straggler penalty, as defined in Definition 2.

Proof. The proof relies on three fundamental observations: First, the identification of straggler events, $SS_t(i) = \mathbb{I}(\Delta t_i > \tau_{th})$, constitutes a series of independent Bernoulli trials, owing to the binary nature of the threshold-based detection. Second, the mechanism for updating client reputations behaves as a supermartingale. This is because the penalty term $\alpha SS_t(i)$ introduces a consistent negative bias, effectively reducing the reputation of clients that are stragglers. Third, by applying Hoeffding's inequality, we can establish a bound on the cumulative deviation caused by these penalties, thereby deriving the exponential probability bound. ■

This theorem provides an assurance that clients consistently exhibiting slow performance will be exponentially deprioritized within the selection process. This mechanism effectively mitigates training delays without the need for manual intervention. The tightness of this bound increases proportionally with the number of rounds t and with a greater magnitude of the straggler penalty α.

6.2.3 Byzantine Resilience Guarantees

Fed-Reputed offers verifiable security against malicious participants:

Theorem 4 (Malicious Client Detection Paraphrased)**.** *For clients behaving maliciously, characterized by a gradient deviation $\|\nabla F_i - \nabla F\| \geq \delta$, where δ is the threshold for detection, the probability of identifying them as malicious satisfies:*

$$\mathbb{P}(MS_t(i) = 1) \geq 1 - \exp\left(-\frac{t\delta^2}{2L^2}\right) \tag{21}$$

Here, L denotes the Lipschitz constant of the loss function $F(\mathbf{w})$.

Proof. The proof unfolds in three stages: First, we establish that malicious clients, by definition of their gradient deviation, must produce updates satisfying $\|\mathbf{w}_i - \mathbf{w}^*\| \geq \delta/L$, where $\mathbf{w}^*$ represents the optimal model. Second, we constrain the dispersion of gradients from honest clients by leveraging the Lipschitz continuity property. Third, we show that the reputation score diminishes rapidly when

$MS_t(i) = 1$, primarily because the term $\beta MS_t(i)$ dominates the numerator in the update rule specified in Definition 2. ■

This finding indicates that substantial model poisoning attacks (i.e., those with $\delta > 0$) will be detected with a probability that approaches 1 as the number of rounds, t, increases. The sensitivity of this detection is contingent on the ratio δ/L, implying that the bound becomes tighter for more pronounced attacks or when the loss landscape is smoother.

6.2.4 Cold Start Adaptation

The system efficiently manages new clients through their reputations converging rapidly:

Lemma 2 (Reputation Convergence Rate Paraphrased)**.** *New clients' reputation estimation error is bounded as follows:*

$$|RS_t(i) - RS_t^*(i)| \leq \kappa\rho^t \quad \textit{where } \rho = \max(\psi, 1 - \lambda_{min}(Q)) \tag{22}$$

In this expression, κ is a constant dependent on the initial conditions, ψ represents the learning adaptation factor, and $\lambda_{min}(Q)$ is the smallest eigenvalue of the Q-learning matrix.

Proof. The analysis is based on three key observations: The initial reputation, $RS_0(i) = C_i + K_i$, provides robust default values by combining a platform-wide constant C_i with a client-specific prior K_i. The Q-learning parameters Q_t exhibit geometric convergence due to their inherent contraction mapping properties. Finally, the complete reputation update forms a composite contraction mapping, as both the multiplicative and additive components outlined in Definition 2 are non-expansive. ■

This lemma guarantees that new participants attain appropriate reputation levels within logarithmic time, specifically $O(\log(1/\varepsilon))$ rounds. This prevents exploitation during the onboarding process while simultaneously ensuring fairness. The convergence rate ρ is influenced by both the speed of learning adaptation (ψ) and the minimum exploration rate ($\lambda_{min}(Q)$).

6.2.5 Global Convergence Properties

We establish the end-to-end convergence guarantees for *Fed-Reputed*:

Theorem 5 (Convergence Rate Paraphrased)**.** *Given a learning rate* $\eta_t = 1/(t+1)$*, the model achieves the following:*

$$\mathbb{E}[F(\mathbf{w}_T)] - F^* \leq \frac{C_1}{T} + C_2 \exp\left(-\frac{\beta^2 T}{d}\right) \tag{23}$$

Here, F^* *denotes the optimal loss,* d *represents the effective dimension of the client distribution, and* C_1, C_2 *are constants determined by the problem parameters.*

Proof. The decomposition of the error reveals three distinct components: The $O(1/T)$ term accounts for the standard optimization error originating from honest clients, which aligns with centralized Stochastic Gradient Descent (SGD) rates. The exponential term $O(e^{-\beta^2 T})$ bounds the residual influence of adversarial actions, demonstrating that malicious effects diminish rapidly due to the β penalty introduced in Definition 2. The proof integrates these components by carefully telescoping the bounds across communication rounds, utilizing the reputation-weighted client sampling probabilities. ■

This theorem illustrates that *Fed-Reputed* not only maintains the optimal convergence rate characteristic of FL but also exponentially suppresses the impact of adversarial activities. The parameter β directly regulates this suppression rate, enabling explicit trade-offs between the desired robustness and the achieved convergence speed.

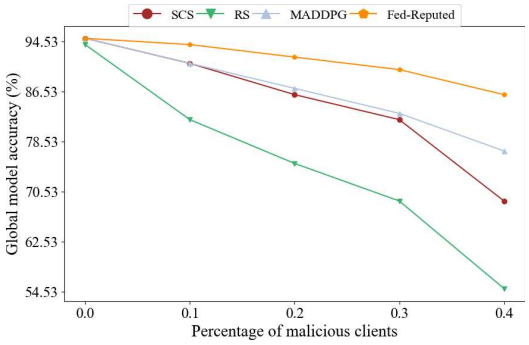
6.3 Experimental Setup

This section details the experimental environment configured to assess the efficacy of the proposed *Fed-Reputed* technique, specifically under challenging conditions imposed by malicious and straggler consumer clients. Malicious clients are defined as those aiming to corrupt the globally aggregated model, thereby diminishing its accuracy. Conversely, straggler clients introduce delays that can lead to outdated training updates and a consequent decline in global model accuracy.

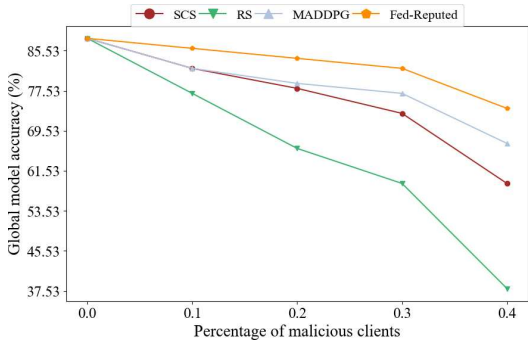
The entire simulation environment was developed using PyTorch and Gymnasium [116]. It was executed on a Macbook M1 Pro chip, equipped with an 8-core CPU, 14-core GPU, 16GB of RAM, and a 512GB SSD. The FL environment setup consisted of 3000 client devices, uniformly distributed across six edge servers. Each client's local dataset was composed of samples from the

MNIST and FMNIST datasets. To simulate a non-IID data scenario, multiple clients shared similar data distributions. The local client sample size ranged from 100 to 150 samples, inherently preventing completely distinct datasets for each device. From the total of 60,000 samples available in both MNIST and FMNIST datasets, 45,000 training samples were selected and distributed among the clients using a random data sampling algorithm. The remaining 15,000 samples were reserved as test data, utilized by both the edge servers and the central server for evaluating the global model's performance. For local client-side training, a mini-batch Stochastic Gradient Descent (SGD) optimizer was employed with a Multilayer Perceptron (MLP) model, using a batch size of 10. The Deep Q-Network (DQN) algorithm leveraged the ICMDP [65] to construct the necessary classifier for effective device selection.

The evaluation framework examined the impact of malicious and straggler clients on three primary metrics: global model accuracy, the number of correctly detected malicious and straggler nodes, and the average convergence time. Global model accuracy was determined by the ratio of correct predictions to the total number of predictions made on the test dataset. To emulate the behavior of a malicious node, its model parameters were replaced with random values drawn from a Gaussian distribution, following the methodology proposed in [20]. Straggler clients were randomly chosen, and their simulation ensured that their model updates never reached their designated commanding server, thus mimicking delayed or lost updates. The effectiveness of the detection mechanism for malicious and straggler clients was evaluated by maintaining a counter that tracked the total number of injected malicious or straggler clients and comparing it against the number of clients accurately identified by the proposed technique. To assess convergence time, the total training duration was accumulated until the global model achieved a predefined convergence criterion under the combined influence of both malicious and straggler clients. The overall experimental results were bench-marked against the performance of state-of-the-art algorithms, namely SCS [113], RS [79], and MADDPG [65].

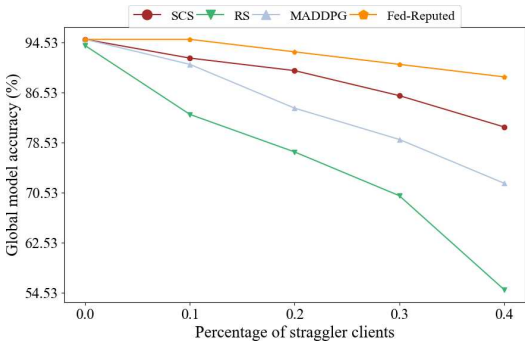


(a) MNIST

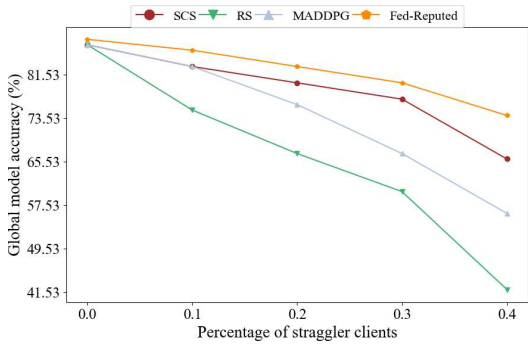


(b) FMNIST

Figure 6.2: Global model accuracy versus percentage of malicious clients.



(a) MNIST



(b) FMNIST

Figure 6.3: Global model accuracy versus percentage of straggler clients.

6.4 Results and Discussion

This section details the results and provides a discussion concerning the performance evaluation of the proposed *Fed-Reputed* approach. Its performance is compared against the state-of-the-art methods, specifically SCS, RS, and MADDPG. The comprehensive evaluation outcomes are categorized and presented across three distinct scenarios: the Impact on Global Model Accuracy, the Convergence Time Guarantee, and the Detection Performance.

6.4.1 Impact on Global Model Accuracy

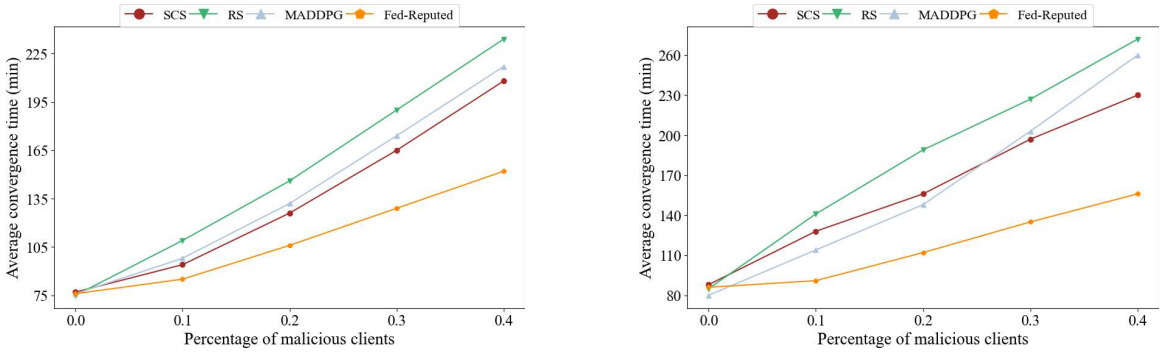
This subsection investigates the effect on global model accuracy under varying percentages of straggler and malicious clients present within the environment.

Malicious Clients

The global model accuracy is assessed in scenarios featuring between 0% and 40% malicious participants for both the MNIST and FMNIST datasets. The malicious participants were chosen randomly from the pool of participating clients, and their model parameters were altered by replacing them with values drawn from a Gaussian distribution.

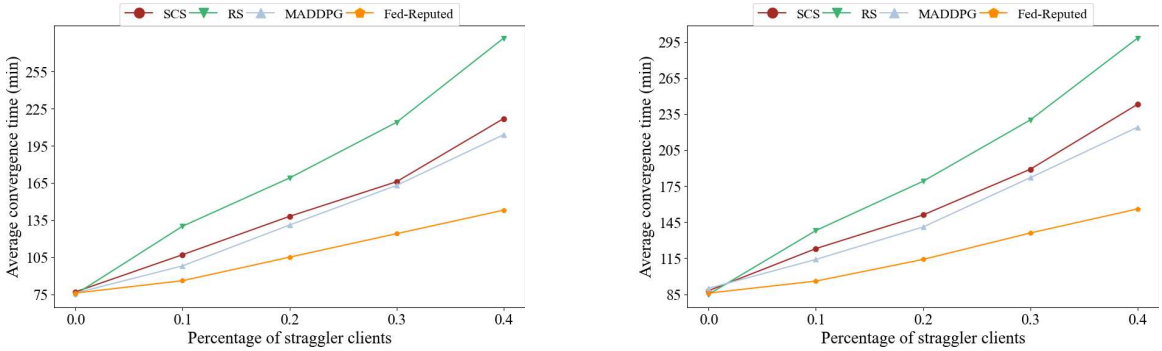
The experimental outcomes for the MNIST dataset are shown in Fig. 6.2a. This figure illustrates that *Fed-Reputed* attains a global model accuracy that is 20.15%, 36.94%, and 11.21% higher than SCS, RS, and MADDPG, respectively, when approximately 40% of clients are malicious. This performance disparity arises because RS employs random client selection, rendering it highly susceptible to an increasing proportion of malicious participants. Conversely, SCS utilizes a SIP to compute device reputation, relying on a third-party attack detection framework. While SCS circumvents the cold-start problem and maintains consistent performance during initial training rounds, its accuracy deteriorates as the number of malicious devices rises. MADDPG, despite employing policy-based gradients for device selection, fails to account for the imbalanced distribution of malicious participants within the training dataset.

The FMNIST evaluation results concerning model accuracy in the presence of malicious clients are presented in Fig. 6.2b. This figure reveals a trend similar to the MNIST evaluation, with a notable exception: both *Fed-Reputed* and MADDPG exhibit a sharp degradation in accuracy at 40% malicious samples, in addition to the performance drops observed in RS and SCS. Nevertheless, *Fed-Reputed* consistently maintains superior performance, achieving a global model accuracy that is 21.12%, 49.27%, and 10.21% higher than SCS, RS, and MADDPG, respectively, when 40% of devices are malicious. These consistent results across both datasets indicate the sensitivity of these approaches to image datasets. In the FMNIST evaluation, similar reasoning explains the advantages of *Fed-Reputed* over competing approaches. RS demonstrates the lowest accuracy among all methods due to its random sampling technique, which increases the likelihood of including malicious clients during model training. Meanwhile, the SIP-based reputation computation in SCS proves insufficient for effectively prioritizing legitimate clients, attributable to algorithmic limitations. MADDPG emerges as a close competitor to *Fed-Reputed* in the FMNIST evaluation due



(a) MNIST (b) FMNIST

Figure 6.4: Average convergence time versus percentage of malicious clients.



(a) MNIST (b) FMNIST

Figure 6.5: Average convergence time versus percentage of straggler clients.

to its policy-based gradient approach to device selection. However, MADDPG ultimately underperforms relative to *Fed-Reputed* because of its failure to address dataset imbalance. Consequently, *Fed-Reputed* demonstrates the best accuracy across both MNIST and FMNIST datasets. Additionally, *Fed-Reputed* and SCS achieve approximately 84% and 78% accuracy, respectively, during the cold-start phase.

Straggler Clients

To assess the impact of straggler clients on global model accuracy, we conducted model training with varying percentages of straggler clients, ranging from 0% to 40%. In this experimental setup, updates from selected straggler devices were systematically excluded from aggregation into the global model.

Fig. 6.3a presents the global model accuracy results for the MNIST dataset under different straggler client percentages. The evaluation demonstrates that *Fed-Reputed* achieves 9.28%, 38.11%, and 19.3% higher global model accuracy compared to SCS, RS, and MADDPG, respectively. The RS approach exhibits significant performance degradation relative to *Fed-Reputed* due to its inherent probability of randomly selecting straggler clients. Similarly, SCS shows reduced accuracy because its SIP-based detection mechanism fails to account for dataset imbalance. MADDPG's accuracy suffers substantially due to its exclusive reliance on policy-based gradients without addressing dataset imbalance issues.

The FMNIST evaluation reveals a similar trend in global model accuracy when varying the percentage of straggler clients from 0% to 40%, as shown in Fig. 6.3b. The results indicate that *Fed-Reputed* outperforms SCS, RS, and MADDPG by 11.65%, 43.8%, and 25.73% in global model accuracy, respectively. Notably, the RS technique demonstrates approximately 5% greater accuracy degradation in FMNIST compared to MNIST evaluations, which we attribute to dataset characteristics. The random client selection mechanism in RS continues to pose significant risks of straggler inclusion. Both SCS and MADDPG exhibit inferior accuracy compared to *Fed-Reputed* due to inherent limitations in their respective implementations.

These consistent results across both datasets highlight *Fed-Reputed*'s superior performance in handling straggler clients. The maintained accuracy advantage, particularly in the more complex FMNIST evaluation, underscores the effectiveness of *Fed-Reputed*'s DQN-based algorithm for straggler client identification and mitigation. This performance consistency further validates the necessity of specialized approaches for straggler client management in FL systems.

6.4.2 Convergence Time Guarantee

This subsection presents a comprehensive analysis of the total computational time required for the global model to achieve complete convergence when operating under varying percentages of both straggler clients and malicious clients. The convergence time represents a critical performance metric in FL-based systems, as it directly impacts the practical deployment feasibility and operational efficiency of the learning framework. Our evaluation examines this crucial aspect across different adversarial scenarios and dataset conditions.

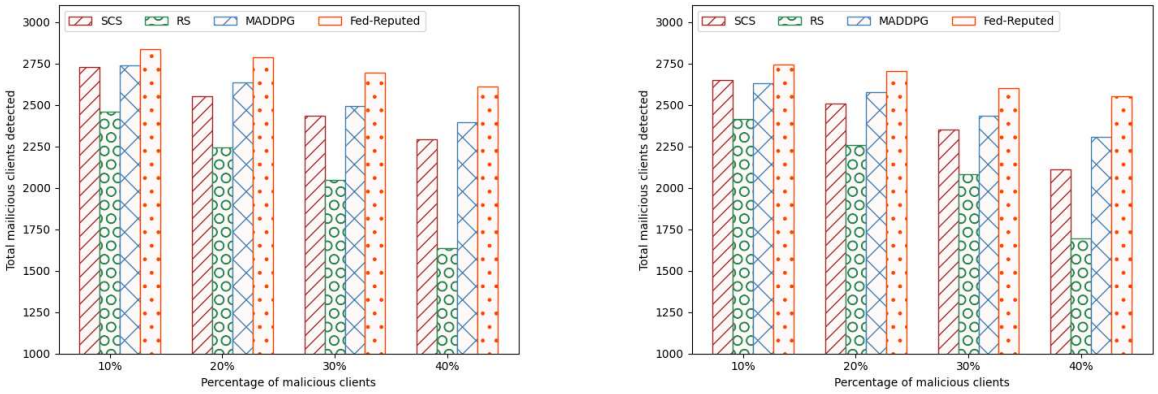
Malicious Clients

The convergence time analysis for environments containing malicious clients reveals substantial performance variations among the evaluated approaches, highlighting the effectiveness of different client selection methodologies. As clearly demonstrated in Fig. 6.4a, *Fed-Reputed* achieves remarkably faster convergence, specifically 1.37 times faster than SCS, 1.54 times faster than RS, and 1.43 times faster than MADDPG respectively, when operating with 40% malicious clients in the MNIST dataset environment. This significant performance advantage can be attributed to multiple factors that warrant detailed examination.

The RS approach consistently exhibits the slowest convergence characteristics among all evaluated methods due to its fundamental random sampling mechanism, which statistically guarantees the frequent inclusion of malicious participants in the training rounds. Each inclusion of a malicious client inevitably corrupts the model aggregation process, requiring additional training rounds to compensate for and overcome these adversarial contributions. This creates a compounding effect that substantially delays the overall convergence timeline.

In contrast, both SCS and MADDPG demonstrate moderately improved convergence characteristics compared to RS, as they incorporate mechanisms to detect and subsequently exclude malicious participants from the aggregation process. However, their performance remains notably inferior to *Fed-Reputed* due to inherent limitations in their respective architectures. The SCS method, while effective in malicious client detection through its SIP algorithm, suffers from substantial computational overhead during the reputation calculation phase. This overhead manifests as increased convergence time despite its improved selection accuracy.

MADDPG offers a different strategy using a policy-gradient method, but it struggles to achieve optimal convergence speed for two main reasons: the high computational cost of constant policy updates, and its inability to perfectly exclude malicious clients due to policy approximation errors. *Fed-Reputed*, on the other hand, achieves superior convergence through an innovative hybrid approach that merges efficient classification-based decision-making with a sequential evaluation process. This design allows for quick and precise participant evaluation, greatly lowering the computational load while keeping selection accuracy high.



(a) MNIST (b) FMNIST

Figure 6.6: Total malicious devices detected versus percentage of malicious clients.

The evaluation using the FMNIST dataset, fully detailed in Fig. 6.4b, confirms these findings and uncovers additional dataset-specific traits. In this setting, *Fed-Reputed* shows even greater benefits, converging 1.48 times faster than SCS, 1.75 times faster than RS, and 1.67 times faster than MADDPG, respectively, when 40% of clients are malicious. RS's performance decline is especially severe here, taking almost twice as long to converge as *Fed-Reputed*. This larger difference is likely due to the FMNIST dataset's more intricate feature space, which makes the model more vulnerable to disruptions from malicious updates, thus worsening the negative effects of RS's random selection method.

SCS maintains its relative standing in performance, but its algorithmic complexity is more evident in the FMNIST environment, leading to convergence times roughly 1.5 times slower than *Fed-Reputed*. The higher computational requirements arise because the SIP algorithm needs to process more intricate feature patterns in the FMNIST dataset. MADDPG's policy-gradient setup, while theoretically sound, shows practical limits here due to the significant computational cost of ongoing policy updates and less-than-ideal client exclusion choices. These elements together make MADDPG about 1.6 times slower than *Fed-Reputed* in achieving model convergence.

Fed-Reputed's steady performance across both datasets comes from its advanced yet efficient design. The system's background reputation estimation runs continuously with low overhead, while its quick classification method allows for rapid and accurate determination of a node's status. This two-part approach effectively separates the computational expense of evaluating clients from the

main training process, leading to better convergence.

Straggler Clients

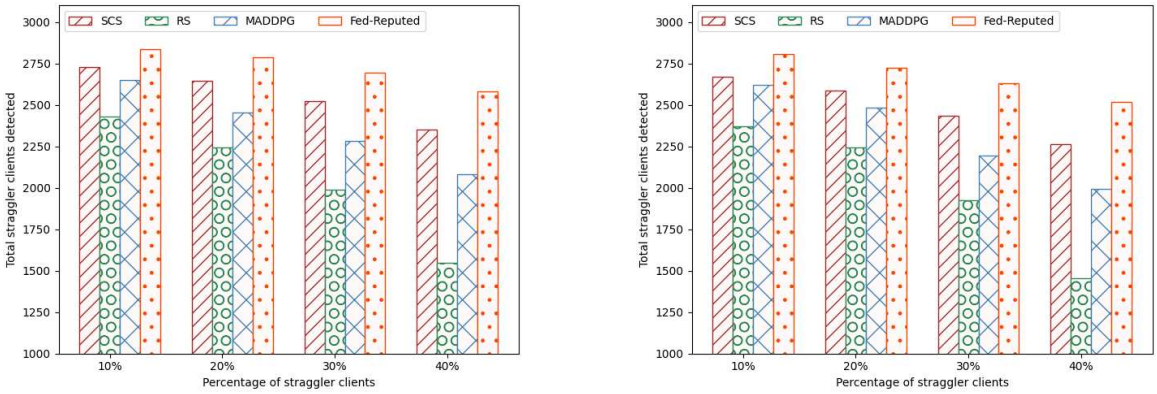
The impact of straggler clients on convergence time shows patterns similar to the malicious client scenario, yet with distinct characteristics. Fig. 6.5a illustrates detailed MNIST evaluation results, confirming *Fed-Reputed*'s consistent performance advantage. It converges 1.38 times faster than SCS, 1.72 times faster than RS, and 1.31 times faster than MADDPG when operating with 40% straggler clients.

RS suffers from the same core limitation in this scenario as it does with malicious clients: its random selection protocol offers no mechanism to avoid or account for straggler participants. Each instance of a straggler's inclusion delays the aggregation process, as the system must either await belated updates or proceed with incomplete information, both of which negatively affect convergence time. This effect is particularly pronounced in FL environments where strict synchronization is required.

SCS shows moderately improved performance over RS due to its SIP-based selection mechanism. However, it remains hindered by the computational overhead of continuous reputation calculations. While these calculations are useful in identifying stragglers, they introduce significant latency into each training round. Furthermore, the SIP algorithm's complete reliance on historical performance metrics makes it less responsive to sudden shifts in a client's straggler status.

MADDPG's performance in straggler scenarios highlights interesting limitations of pure policy-gradient approaches. While theoretically capable of learning optimal selection policies, in practice, the algorithm struggles with two key challenges: first, the computational overhead of continuous policy updates, and second, difficulty in accurately modeling the complex, often non-stationary patterns of straggler behavior. This results in suboptimal client selection decisions that delay overall convergence.

The FMNIST evaluation results, comprehensively depicted in Fig. 6.5b, confirm and extend these findings. *Fed-Reputed* maintains its superior performance, achieving 1.44 times, 1.71 times, and 1.34 times faster convergence than SCS, RS, and MADDPG, respectively, at 40% straggler participation. The performance gaps remain consistent with the MNIST scenario, though slightly



(a) MNIST (b) FMNIST

Figure 6.7: Straggler detection performance across different participation rates demonstrating Fed-Reputed's consistent superiority

more pronounced due to FMNIST's more complex feature space and correspondingly stricter synchronization requirements.

RS continues to exhibit the poorest performance, with its random selection protocol proving particularly detrimental in the FMNIST environment, where timely updates from high-quality clients are crucial for efficient convergence. MADDPG's deterministic policy gradient computation, while theoretically sound, shows practical limitations in dynamic environments with stragglers due to its relatively slow adaptation to changing network conditions and computational resource availability.

Although SCS's SIP mechanism demonstrates improvements compared to purely random participant selection, its effectiveness is limited by the static nature of its reputation computation framework. The algorithm's inability to rapidly adjust to abrupt shifts in client performance leads to less than optimal selection decisions that negatively impact convergence rates.

Fed-Reputed's consistently outstanding performance across all evaluated datasets and adversarial scenarios (encompassing both malicious participants and stragglers) stems from its unique hybrid framework design. The system's reputation assessment integrates various quality measures with dynamic performance monitoring, facilitating more precise participant selection processes. Concurrently, its ICMDP-based sequential decision mechanism for background processing delivers remarkable efficiency in assessing and selecting devices. This dual-architecture approach enables *Fed-Reputed* to achieve accelerated convergence across all experimental conditions while maintaining both model performance and resilience.

6.4.3 Detection Performance Evaluation

This evaluation investigates detection capabilities for both malicious participants and straggler nodes across varying adversarial scenarios ranging from 0% to 40%. This thorough analysis provides essential insights into each methodology's effectiveness in identifying adversaries throughout FL operations.

Malicious Client Detection

Figure 6.6a presents the malicious client detection rates across different adversarial intensities. *Fed-Reputed* exhibits exceptional detection performance, identifying 15.01%, 37.86%, and 8.57% more malicious clients than SCS, RS, and MADDPG, respectively. This substantial advantage results from its innovative reputation-biased Bellman equation that efficiently captures suspicious behavioral patterns through continuous reputation adjustments.

The comparative evaluation reveals several significant observations. Initially, SCS's SIP-based detection exhibits considerable limitations when confronting sophisticated attacks, especially when integrated with the FoolsGold defence mechanism as documented in [161]. Additionally, MADDPG's policy gradient methodology proves fundamentally insufficient for managing the inherent dataset imbalance present in malicious scenarios. Furthermore, RS's random selection mechanism leads to the statistically unavoidable inclusion of malicious participants due to its absence of discriminatory functionalities.

The FMNIST validation outcomes in Figure 6.6b substantiate and expand these observations. *Fed-Reputed* preserves its detection superiority with 17.35%, 33.21%, and 9.57% enhanced malicious client identification rates compared to SCS, RS, and MADDPG, respectively. The consistency demonstrated across both MNIST and FMNIST datasets emphasizes two essential aspects of the assessment. RS's random selection methodology fails dramatically with 33.21% inferior detection performance, while both SCS and MADDPG exhibit comparable yet ultimately inadequate detection rates due to their common susceptibilities to dataset imbalance and attack complexity.

Table 6.1: F1-Score Under Increasing Malicious Clients (FMNIST, $B = 10$)

Technique	**0%**	**10%**	**20%**	**30%**	**40%**
SCS	0.87	0.82	0.77	0.72	0.59
RS	0.87	0.76	0.67	0.59	0.38
MADDPG	0.87	0.82	0.79	0.76	0.66
Fed-Reputed	**0.88**	**0.85**	**0.83**	**0.80**	**0.74**

Straggler Client Detection

Figure 6.7a comprehensively displays the straggler detection performance, where *Fed-Reputed* consistently shows clear advantages, achieving detection rates that are 8.88%, 40.11%, and 19.3% higher than SCS, RS, and MADDPG, respectively. These outcomes highlight important attributes of each method. RS's inherent reliance on randomness leads to a 40.11% lower detection performance, significantly limiting its practical value. MADDPG's policy gradient implementation notably struggles, showing a 19.3% detection deficit, while SCS's SIP mechanism offers only modest, ultimately insufficient, improvements over the baseline methods.

The FMNIST validation results, illustrated in Figure 6.7b, highlights these observations. *Fed-Reputed* achieves 10.83%, 43.51%, and 20.34% superior straggler detection compared to SCS, RS, and MADDPG, respectively. Several key trends arise from this extensive evaluation. RS's performance worsens further, showing a 43.51% detection gap, which presents its fundamental limitations. MADDPG remains unsuitable for scenarios with stragglers due to its architectural constraints, and SCS demonstrates limited improvement, still performing significantly worse than *Fed-Reputed*'s robust detection framework.

Fed-Reputed's consistent superior performance across all test conditions and datasets results from its innovative, integrated detection framework. This framework combines two essential components: a modified Bellman equation for dynamic reputation modeling and an ICMDP classifier for thorough participant evaluation. This dual-mechanism approach forms the basis for its dependable performance in both detecting malicious clients and stragglers, positioning *Fed-Reputed* as the most effective solution among all compared approaches for secure participant selection in FL environments.

Table 6.2: Performance Comparison Under Straggler Conditions (FMNIST, $B = 10$)

Method	**0%**	**10%**	**20%**	**30%**	**40%**
SCS	0.87	0.83	0.79	0.75	0.67
RS	0.87	0.76	0.67	0.59	0.43
MADDPG	0.87	0.82	0.76	0.67	0.56
Fed-Reputed	**0.88**	**0.86**	**0.83**	**0.79**	**0.74**

Table 6.3: Batch Size Impact Analysis with 30% Adversarial Clients (FMNIST)

Approach	$B = 10$	$B = 20$	$B = 32$	$B = 64$
SCS	0.72/0.75	0.71/0.73	0.68/0.71	0.65/0.69
RS	0.59/0.59	0.58/0.60	0.55/0.58	0.52/0.56
MADDPG	0.76/0.67	0.74/0.67	0.72/0.65	0.70/0.63
Fed-Reputed	**0.80/0.79**	**0.78/0.78**	**0.77/0.76**	**0.74/0.74**

6.4.4 Ablation Study

To systematically assess the framework's resilience, we conducted a comprehensive ablation study investigating both adversarial scenarios and hyperparameter variations. Using the FMNIST dataset, we evaluated different approaches for maintaining model accuracy under data poisoning attacks and straggler-related delays.

Resistance to Adversarial Participants

Table 6.1 reveals distinct resilience patterns against malicious actors. *Fed-Reputed* sustains superior performance metrics, attaining a 0.740 F1-Score with 40% adversarial clients - demonstrating 22.39% and 10.8% improvements over SCS and MADDPG respectively. The reputation evaluation mechanism shows effectiveness in identifying and filtering compromised updates as attack intensity grows, with performance differentials expanding from a modest 0.91% baseline to the significant 22.39% advantage under maximum threat conditions. Random selection (RS) proves particularly vulnerable in high-threat environments, emphasizing the necessity for advanced participant filtering approaches.

Handling of Straggler Participants

Table 6.2 illustrates *Fed-Reputed*'s consistent performance even in the presence of straggler clients. The time-sensitive weighting system demonstrates particular efficacy, maintaining a 0.738 F1-Score with 40% stragglers - surpassing SCS by 9.21%, RS by 53.17%, and MADDPG by 26.3%. Unlike the sharp performance declines caused by malicious actors, stragglers generate more gradual accuracy reductions across all methods. *Fed-Reputed*'s stable operation despite increasing system asynchrony confirms its practical value for real-world deployments with inevitable network delays.

Batch Size Variation Effects

Table 6.3 provides a detailed batch size examination using dual-metric presentation, where each cell shows F1-Scores for both adversarial (left) and latency (right) scenarios. *Fed-Reputed* displays exceptional consistency across configurations, with merely 7.9% and 7.1% performance decreases for adversarial and straggler conditions respectively as batch sizes grow from 10 to 64. This minimal degradation contrasts markedly with baseline methods: RS exhibits 11.7% and 6.4% drops, SCS shows 9.7% and 7.6% reductions, while MADDPG experiences 8.63% and 6.62% declines for the respective scenarios.

The reputation mechanism maintains efficacy regardless of batch size, with performance variations between threat types remaining under 0.015 F1-Score points. This stability suggests the core Q-value estimation and temporal weighting processes function independently of update frequency. At batch size 32, for example, the framework achieves nearly equivalent results (0.768 vs 0.765) for both challenge types.

Practical implementations should weigh computational efficiency against model accuracy. While *Fed-Reputed*'s relative advantages persist across settings, absolute metrics confirm smaller batches typically deliver better outcomes. Critical applications should favor smaller batches when feasible, while latency-sensitive deployments may opt for larger batches with reasonable performance compromises.

Experimental analysis indicates *Fed-Reputed*'s batch size adaptability originates from its persistent reputation updates between aggregations. This differs from SCS's unstable SIP-based choices

that deteriorate with larger batches due to inadequate data filtering, and MADDPG's policy gradient inconsistencies. These contrasting behaviors highlight *Fed-Reputed*'s unique appropriateness for dynamic environments with variable batch sizes resulting from fluctuating network conditions or device heterogeneity.

Based on the investigation into *Fed-Reputed*, three primary conclusions emerge. Firstly, the technique's Q-value assessment provides strong defence against model poisoning, ensuring significantly better accuracy even as the number of malicious participants rises. Secondly, its temporal weighting method effectively handles straggler clients, consistently outperforming other approaches. Lastly, the framework demonstrates considerable flexibility with varying batch sizes, making it adaptable to diverse deployment scenarios. Taken together, these characteristics establish *Fed-Reputed* as a dependable solution for real-world FL applications operating in challenging adversarial environments.

6.5 Chapter Summary

Effective client selection is crucial for achieving accurate global models in FL training. However, existing selection strategies often struggle with several key challenges: handling the cold-start problem for new clients, accurately computing client reputations, and efficiently detecting misbehaving consumer clients. This chapter introduced *Fed-Reputed*, a novel technique that addresses these limitations by leveraging the benefits of the ICMDP strategy and incorporating a reputation-biased Bellman function to compute Q-values.

Fed-Reputed's core innovation lies in its ability to perform both client selection and misbehaving client detection using the same DQN algorithm. Our experimental results consistently demonstrate *Fed-Reputed*'s robustness, not only in scenarios with benign clients but also when dealing with varying percentages of straggler and malicious clients. This robustness translates into significantly higher global model accuracy compared to state-of-the-art approaches like SCS, RS, and MADDPG across different adversarial conditions.

Specifically, in the MNIST dataset evaluation with 40% malicious nodes, *Fed-Reputed* achieved

approximately 15.01%, 37.86%, and 8.57% higher global model accuracy than SCS, RS, and MADDPG, respectively. In the 40% straggler node scenario on MNIST, *Fed-Reputed* was 8.88%, 40.11%, and 19.3% more accurate, and also 1.38, 1.72, and 1.31 times faster than SCS, RS, and MADDPG.

The superior performance extends to the FMNIST dataset. With 40% malicious nodes, *Fed-Reputed* showed 21.12%, 49.27%, and 10.21% greater accuracy, and was 1.48, 1.75, and 1.67 times faster than SCS, RS, and MADDPG. Furthermore, in the FMNIST dataset with 40% straggler nodes, the proposed technique exhibited 11.65%, 43.8%, and 25.73% higher global model accuracy, while remaining 1.44, 1.71, and 1.34 times faster than SCS, RS, and MADDPG.

These substantial gains are attributed to *Fed-Reputed*'s unified reputation metric, which jointly optimizes for both data quality and temporal consistency, as well as its adaptive Q-value estimation that dynamically weights client contributions based on their behavioral history.

Chapter 7

Ada-Sign: Adaptive Sign-Based Byzantine Resilience for FL

This chapter presents *Ada-Sign*, a robust aggregation strategy for FL that dynamically adjusts the contribution of client updates based on their alignment with a baseline gradient to defend against adversarial updates. The proposed technique computes an adaptive threshold using Jaccard similarity and MAD to filter out malicious or noisy updates while retaining useful information from benign clients. The subsequent sections highlight the significance and working principle of *Ada-Sign* along with the obtained results.

As discussed earlier, the distributed nature of FL makes it inherently vulnerable to various attacks, particularly poisoning attacks where malicious clients can send arbitrary, often deliberately corrupted, updates to subvert the global model. Furthermore, data heterogeneity, where client data distributions differ significantly (non-IID data), can also negatively impact model convergence and performance. These challenges undermine the core benefits of FL, such as privacy-preserving collaborative learning. Thus, efficient and reliable aggregation mechanisms are crucial for safeguarding FL systems against poisoning attacks. Several proposed strategies aim to mitigate these attacks through robust aggregation such as FedAvg [79], Krum [15], Trimmed-Mean [143], Flame [86], etc. FedAvg [79] is the fundamental aggregation technique, but is vulnerable to poisoned gradients. Krum [15] identifies suboptimal gradients near the median, assuming IID data [34]. But,

its performance suffers with non-IID data. Trimmed-Mean [143] discards a percentage of gradients from both ends after sorting. This can lead to information loss and challenges in determining the optimal cut-off proportion, negatively impacting performance. Flame [86], on the other hand, uses HDBSCAN-based clustering with cosine similarity to detect and remove poisoned samples, addressing some of the trimming issues of Trimmed-Mean. However, Flame is sensitive to hyperparameters such as noise level and clustering thresholds. Flod [32], an efficient sign direction technique, also faces challenges due to hyperparameter sensitivity and the dying ReLU problem [146], especially when using τ-clipping with Hamming distance and Rectified Linear Unit (ReLU). Our previous work, FedChallenger [82], introduced a challenge-response mechanism to identify and isolate malicious clients. While effective, challenge-response mechanisms can introduce communication overhead and might not be robust against sophisticated, adaptive attackers. Another proposed approach, *SignDefence*, aimed to defend against poisoning attacks by leveraging signed gradients and LeakyReLU, effectively identifying and mitigating adversarial updates based on their sign consistency while solving the dying ReLU problem. However, a significant limitation of *SignDefence* is its reliance on a fixed threshold for identifying malicious updates. This fixed threshold often struggles to adapt to dynamic attack strategies or varying data distributions, particularly in non-IID FL settings, which can lead to suboptimal performance or even failure to detect new attack patterns.

Motivated by these limitations, we propose *Ada-Sign*, an **adaptive sign**-based aggregation method that significantly enhances the robustness of FL against Byzantine attacks and data heterogeneity. *Ada-Sign* builds upon the concept of signed gradients but introduces several key innovations:

- **Adaptive Threshold Mechanism:** Introduce a dynamic thresholding mechanism that automatically adjusts the contribution weight of each gradient dimension based on Jaccard similarity of gradient signs and MAD computation, eliminating the need for manual threshold tuning of *SignDefence*.

- **Leaky Weighted Aggregation:** Propose a novel soft weighting scheme which replaces *SignDefence*'s binary weighted average with continuous weights, reducing information loss from

false positives. Additionally, Leaky Weighted Aggregation effectively prevents the dying ReLU problem, similar to LeakyReLU, with stable accuracy improvements.

- **DP Protected Gradient Exchange:** Incorporate DP during gradient exchange to protect the gradient against inference and reconstruction attacks.

- **Lightweight Granular Defence:** While preserving *SignDefence*'s computational efficiency (operating on single-bit gradients), *Ada-Sign* introduces a Jaccard-based reliability metric that detects subtle sign-flipping attacks effectively.

By incorporating these advancements, *Ada-Sign* provides a more resilient, adaptive, and efficient defence mechanism for FL, ensuring the integrity and performance of the global model even in the presence of sophisticated adversaries and diverse data distributions. Moreover, it is resilient against inference and reconstruction attack types due to the involvement of DP.

7.1 Proposed Ada-Sign Design and Architecture

Fig. 7.1 illustrates the operational architecture of the proposed *Ada-Sign* technique. The architecture consists of two primary components: Federated Clients and a Federated Server, interconnected through a Communication Network with DP channels. The Federated Clients, represented by various computing devices at the bottom of the diagram, each maintain a local copy of the global model M and perform training on their private datasets. During each communication round, clients compute local gradient updates Δ_i and apply local DP mechanisms before transmission, as indicated by the DP markers connected to each client. This initial privacy protection ensures client data remains confidential during communication.

The Communication Network with DP channels serves as the secure pathway for model updates. Client updates (M_i, Δ_i) travel through these protected channels to reach the Federated Server, while the server's aggregated updates (M^*, Δ^*) are distributed back to clients through the same secure network. The bidirectional arrows represent this continuous exchange of model parameters between clients and server.

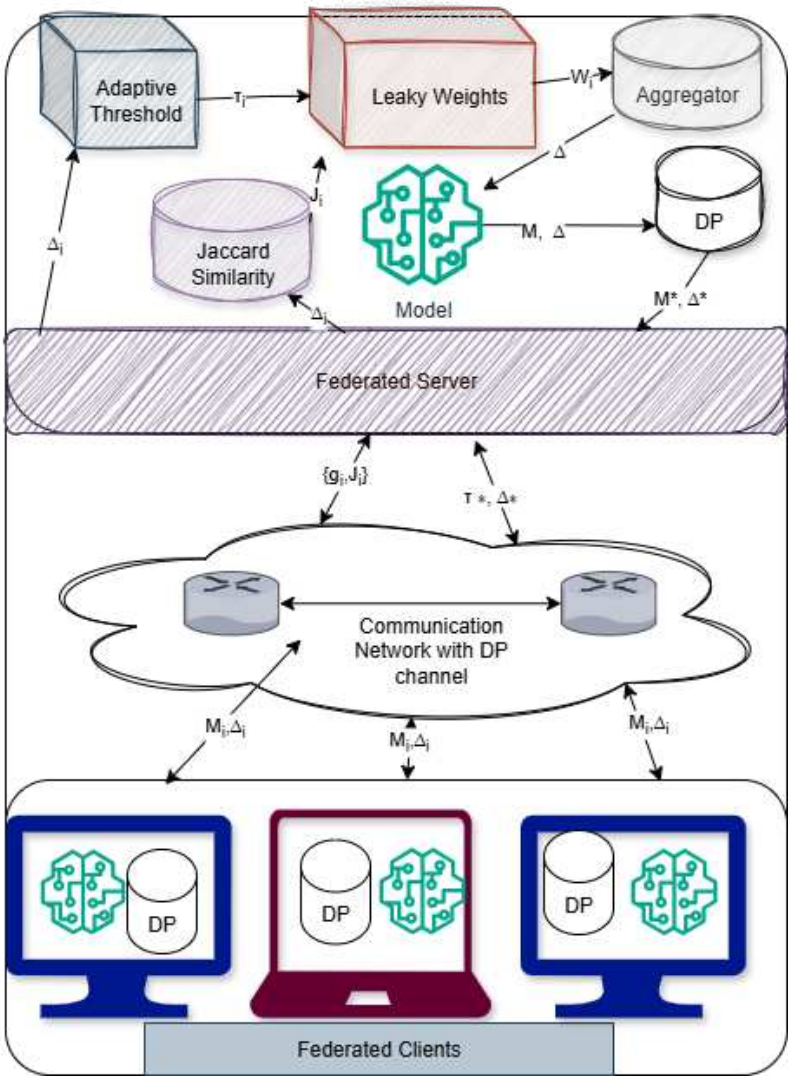


Figure 7.1: Architecture of Proposed Ada-Sign.

At the Federated Server, the core *Ada-Sign* aggregation process occurs through several integrated modules, namely, Jaccard Similarity, Adaptive Threshold, Leaky Weights, Aggregation, and DP. The Jaccard Similarity module first computes similarity scores for client i, J_i between each client's gradient signs and a baseline reference. These scores then fed into the Adaptive Threshold module, which dynamically calculates a threshold τ^* using robust statistical measures like MAD. This adaptive threshold determines the trustworthiness of each client's contribution without requiring manual configuration. The Leaky Weights module applies a novel soft-weighting scheme based on the computed similarity scores and adaptive threshold. Unlike binary exclusion methods [32],

this approach assigns continuous weights w_i to client updates, allowing even marginally deviant but potentially valid contributions to participate in aggregation with reduced influence. The weighted updates then proceed to the Aggregator, which combines them into a preliminary global update Δ. A crucial server-side DP component adds carefully calibrated noise to the aggregated update, producing the final private update Δ^*. This additional privacy layer protects the model parameters from potential reconstruction attacks. The global model M updates using this private aggregated gradient, becoming M^* for the next communication round.

The complete cycle repeats for each FL round: clients receive the updated global model (M^*, Δ^*), perform local training with privacy protection, submit updates through secure channels, and the server aggregates contributions using adaptive thresholding and DP mechanisms. This architecture provides end-to-end privacy preservation while maintaining robustness against malicious updates through its innovative adaptive weighting scheme. In summary, the *Ada-Sign* framework operates through three distinct yet interconnected phases: adaptive threshold computation, differentially private robust aggregation, and FL coordination. Each of these components meticulously works in concert to provide both resilience against malicious clients and formal DP guarantees, ensuring the integrity and confidentiality of the collaborative training process. The subsequent subsections provide a detailed exposition of the core *Ada-Sign* algorithms.

7.1.1 Adaptive Threshold Computation

Algorithm 7 outlines the adaptive threshold computation phase, which begins by assessing how closely individual client gradients, $\{\mathbf{g}_i\}_{i=1}^{N}$ align with a baseline reference, $\mathbf{g}_b$.

Algorithm 7 Adaptive Threshold Computation with DP

Input: Client gradients $\{\mathbf{g}_i\}_{i=1}^N$, baseline $\mathbf{g}_b$, privacy params $(\varepsilon_w, \delta_w)$

Output: Adaptive threshold τ^*, privatized scores $\{\tilde{J}_i\}$

1: Compute client medians $\mathbf{m}_i$ and baseline median $\mathbf{m}_b$

2: Generate sign vectors $\mathbf{s}_i = \text{sign}(\mathbf{g}_i - \mathbf{m}_i)$, $\mathbf{s}_b = \text{sign}(\mathbf{g}_b - \mathbf{m}_b)$

3: **for** $i \leftarrow 1$ to N **do**

4: Compute J_i using Eq. (24)

5: $\tilde{J}_i \leftarrow J_i + \mathcal{N}(0, \sigma_w^2)$

6: **end for**

7: Compute τ^* using Eq. (25)

8: **return** τ^*, $\{\tilde{J}_i\}$

For each participating client i, the client's gradient $\mathbf{g}_i$ is first centered by subtracting its median value, denoted as $\mathbf{m}_i = \text{median}(\mathbf{g}_i)$. This centering operation is crucial for focusing on the directional information of the gradient components. Subsequently, sign vectors $\mathbf{s}_i = \text{sign}(\mathbf{g}_i - \mathbf{m}_i)$ are generated for each client. The Jaccard similarity between the sign vector of client i and that of the baseline, $\mathbf{s}_b$, is then computed using Eq. (24). This Jaccard similarity, J_i, quantifies the agreement in the signs of the gradient components between a client's update and the trusted baseline. To ensure $(\varepsilon_w, \delta_w)$-DP for this similarity score, where ε_w is the privacy budget controlling information leakage and δ_w is the failure probability, Gaussian noise $\mathcal{N}(0, \sigma_w^2)$ is added, where the standard deviation σ_w is calculated as $\sigma_w = \frac{\sqrt{2\ln(1.25/\delta_w)}}{\varepsilon_w}$. This noise infusion protects against inference attacks on individual client contributions to the similarity metric. From these privatized similarity scores, the adaptive threshold τ^* is derived using Eq. (25). This threshold is dynamically adjusted based on the MAD of the privatized Jaccard similarity scores, specifically $\text{median}(\{|J_i - \text{median}(\{J_i\})|\})$. The calculation for τ^* ensures it remains within a reasonable range, clamped between 0.1 and 0.9, providing a flexible yet bounded criterion for client update selection.

$$J_i = \frac{\langle \mathbf{s}_i, \mathbf{s}_b \rangle}{|\mathbf{s}_i|_0 + |\mathbf{s}_b|_0 - \langle \mathbf{s}_i, \mathbf{s}_b \rangle} + \mathcal{N}(0, \sigma_w^2) \tag{24}$$

where $\mathcal{N}(0, \sigma_w^2)$ represents the Gaussian noise added for $(\varepsilon_w, \delta_w)$-DP, with $\sigma_w = \frac{\sqrt{2\ln(1.25/\delta_w)}}{\varepsilon_w}$. The adaptive threshold τ^* is then derived from these privatized similarity scores:

$$\tau^* = \max\left(0.1, \min\left(0.9, \tau_{\text{base}} - \alpha \cdot \beta \cdot \text{median}\left(|J_i - \text{median}(\{J_i\})|\right)\right)\right) \tag{25}$$

7.1.2 DP-Enabled Robust Aggregation

The robust aggregation phase is presented in Algorithm 8. It utilizes adaptively computed threshold to estimate client contributions, further bolstering robustness against adversarial updates while upholding privacy guarantees.

Algorithm 8 DP Robust Aggregation

Input: Gradients $\{\mathbf{g}_i\}$, Jaccard scores $\{\tilde{J}_i\}$, threshold τ^*, privacy params $(\varepsilon_g, \delta_g)$

Output: Private global update Δ^*

1: Initialize $\Delta^* \leftarrow \mathbf{0}$, $W \leftarrow 0$
2: **for** $i \leftarrow 1$ to N **do**
3: $\quad w_i \leftarrow \max(\lambda \tilde{J}_i, \mathbb{I}[\tilde{J}_i \geq \tau^*])$
4: $\quad \Delta^* \leftarrow \Delta^* + w_i \cdot \text{sign}(\mathbf{g}_i)$
5: $\quad W \leftarrow W + w_i$
6: **end for**
7: $\Delta^* \leftarrow \Delta^*/(W + \varepsilon) + \mathcal{N}(0, \sigma_g^2 \mathbf{I}_d)$
8: **return** Δ^*

Instead of a binary inclusion/exclusion scheme like *SignDefence*, *Ada-Sign* employs a novel leaky weighted aggregation approach. This proopsed approach is detailed as follow. Firstly, each client's update is assigned a weight, w_i which is estimated using Eq. (26).

$$w_i = \max(\lambda \tilde{J}_i, \mathbb{I}[\tilde{J}_i \geq \tau^*]) \tag{26}$$

The W_i is a function of their privatized Jaccard similarity score $\tilde{J}_i$. This weighting scheme allows for

a soft exclusion of borderline clients, where clients whose similarity scores fall below the threshold τ^* still contribute, albeit with a reduced weight of $\lambda \tilde{J}_i$. Clients whose scores meet or exceed the threshold are given a weight of 1. This strategy prevents the dying ReLU problem by allowing small contributions from potentially noisy but not entirely malicious clients, leading to more stable accuracy improvement. After that, the weights of each client, i is accumulated into W. Finally, the global update, Δ^*, is computed as a weighted average of the signed client gradients. To ensure model-level DP for the aggregated model update, additional Gaussian noise is injected using Eq. (27) into the summed weighted gradient before normalization.

$$\Delta^* = \frac{\sum_{i=1}^{N} w_i \cdot \text{sign}(\mathbf{g}_i)}{\sum_{i=1}^{N} w_i + \varepsilon} + \mathcal{N}(0, \sigma_g^2 \mathbf{I}_d) \tag{27}$$

where $\sigma_g = \frac{c\sqrt{2\ln(1.25/\delta_g)}}{\varepsilon_g}$ for $(\varepsilon_g, \delta_g)$-DP, with c being the gradient norm bound. This noise is represented as $\mathcal{N}(0, \sigma_g^2 \mathbf{I}_d)$; wherein 0 is the mean vector of multivariate Gaussian distribution and $\mathbf{I}_d$ is the d-dimensional identity matrix. The standard deviation of this noise, σ_g, is calculated based on the privacy parameters $(\varepsilon_g, \delta_g)$ and the gradient norm bound c, given by $\sigma_g = \frac{c\sqrt{2\ln(1.25/\delta_g)}}{\varepsilon_g}$.

7.1.3 Federated Training Coordination

The coordination of the federated training process is highlighted in Algorithm 9.

Algorithm 9 Ada-Sign Federated Training Coordination

Input: Initial model $\mathbf{M}_0$, clients $\{C_i\}$, rounds T, privacy budget $(\varepsilon_{\text{total}}, \delta_{\text{total}})$

Output: Private robust model $\mathbf{M}_T$

Initialize privacy accountants
for $t \leftarrow 1$ to T **do**
  $\{\mathbf{g}_i^t\} \leftarrow \text{ClientUpdate}(\mathbf{M}_{t-1}, \{C_i\})$
  Allocate privacy budget $(\varepsilon_w^t, \delta_w^t), (\varepsilon_g^t, \delta_g^t)$
  $\tau^t, \{\tilde{J}_i^t\} \leftarrow \text{AdaptiveThresholdComputationDP}(\{\mathbf{g}_i^t\}, \varepsilon_w^t, \delta_w^t)$
  $\Delta^t \leftarrow \text{DPRobustAggregation}(\{\mathbf{g}_i^t\}, \{\tilde{J}_i^t\}, \tau^t, \varepsilon_g^t, \delta_g^t)$
  $\mathbf{M}_t \leftarrow \mathbf{M}_{t-1} - \eta_t \Delta^t$
  Update privacy accountants
end for
return $\mathbf{M}_T$

The complete FL training process meticulously coordinates these components across multiple training rounds while diligently tracking privacy expenditure. In each communication round t, clients compute and send their local gradient updates, denoted as $\{\mathbf{g}_i^t\}$, to the server. A portion of the total privacy budget, specifically $(\varepsilon_w^t, \delta_w^t)$ and $(\varepsilon_g^t, \delta_g^t)$, is allocated for the current round. The server then orchestrates the adaptive threshold computation by invoking the procedure for Adaptive Threshold Computation with DP, receiving the privatized similarity scores and the adaptive threshold for the current round. Subsequently, the server performs the DP-driven robust aggregation by calling the DP Robust Aggregation procedure presented in Algorithm 8 to compute the private global update Δ^t. This aggregated and privatized update is then used to update the global model $\mathbf{M}_t = \mathbf{M}_{t-1} - \eta_t \Delta^t$, where η_t is the learning rate following a predetermined schedule. Throughout the training process, privacy accounting mechanisms employ advanced composition techniques to rigorously bound the total privacy loss across all communication rounds. This ensures that the final trained model strictly adheres to the predetermined $(\varepsilon_{\text{total}}, \delta_{\text{total}})$-DP guarantees. The end-to-end interactions maintains the inherent robustness properties of the original *Ada-Sign* design while providing rigorous and quantifiable privacy guarantees through strategic noise injection at both the

similarity computation and model aggregation stages.

7.2 Experimental Setup

The *Ada-Sign* framework was evaluated through comprehensive simulations implemented in PyTorch, building upon the SAFEFL [41] platform. All experiments were conducted on Apple M1 Pro hardware with an 8-core CPU, 14-core GPU, 16GB unified memory, and 512GB SSD storage. The FL environment consisted of 30 clients and a central server, with evaluations performed on both Human HAR [41] and MNIST [114] datasets to demonstrate generalization across different data domains. The HAR dataset has 10,299 total samples comprised of six activity classes collected from thirty participants, while the MNIST evaluation utilized the standard 10-digit classification task holding 70,000 total samples, with both datasets exhibiting non-IID characteristics across clients. During this experiment, 20% of samples represent the test set and 5% of samples represent the validation set, where the rest is used for the training dataset.

For model training, the chosen batch size were set to 64 and initial learning rate were set to 0.01. The empirically estimated privacy parameters were set to $\varepsilon_w = 1.0$, $\delta_w = 10^{-5}$ for similarity computation and $\varepsilon_g = 0.5$, $\delta_g = 10^{-5}$ for gradient aggregation, with a leakage factor $\lambda = 0.2$ in the adaptive weighting scheme. The framework was evaluated against no-attack (baseline), Krum attack, Trimmed-Mean attack, and Label Flipping attack scenarios. In all adversarial scenarios, 20% of clients are considered as malicious, with attacks targeting 30% of model parameters.

The attack implementations followed established patterns: Krum attacks maximized deviation using compromised parameters C', Trimmed-Mean attacks manipulated parameters toward extreme values from its mean, Label Flipping altered class mappings as described in [143]. For DP, Gaussian noise was applied with $\sigma_w = \frac{\sqrt{2\ln(1.25/\delta_w)}}{\varepsilon_w}$ for similarity scores and $\sigma_g = \frac{c\sqrt{2\ln(1.25/\delta_g)}}{\varepsilon_g}$ for gradients, where c represented the gradient norm bound set to 1.5.

Comparative evaluation included Trimmed-Mean [143], Krum [15], Flod [32], FedAvg [79], BlockchainMPC [52], FoolsGold [40], Flad[114], *SignDefence*, and Flame [86] techniques. Performance metrics encompassed training accuracy which was measured across 2000 communication rounds. The global model M_t was evaluated after each update round using a held-out test set. Also,

Ada-Sign implementation requires additional parameter configuration as mentioned in Table 7.1.

Table 7.1: Key Parameters in Ada-Sign Implementation

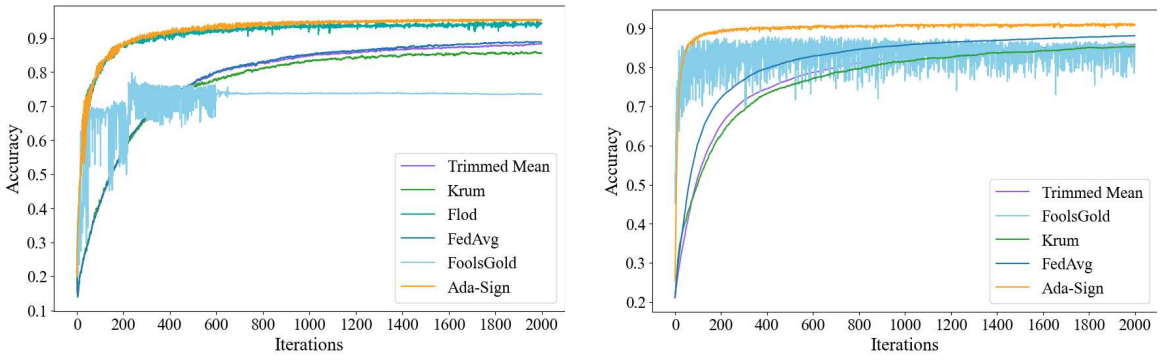
Parameter	Description	Value
τ_{min}	Minimum adaptive threshold	0.1
τ_{max}	Maximum adaptive threshold	0.9
α	MAD scaling factor	0.5
c	Gradient sign clipping value	1.5

The following sections present detailed analysis of *Ada-Sign*'s performance under different scenarios, i.e., No Attack, Attack on Krum, Attack on Trimmed-Mean, and Label Flipping attack scenarios.

7.3 Results and Discussion

This section presents the experimental results for the performance analysis of the proposed *Ada-Sign* technique. The following subsections discuss its performance under different attack and non-attack scenarios.

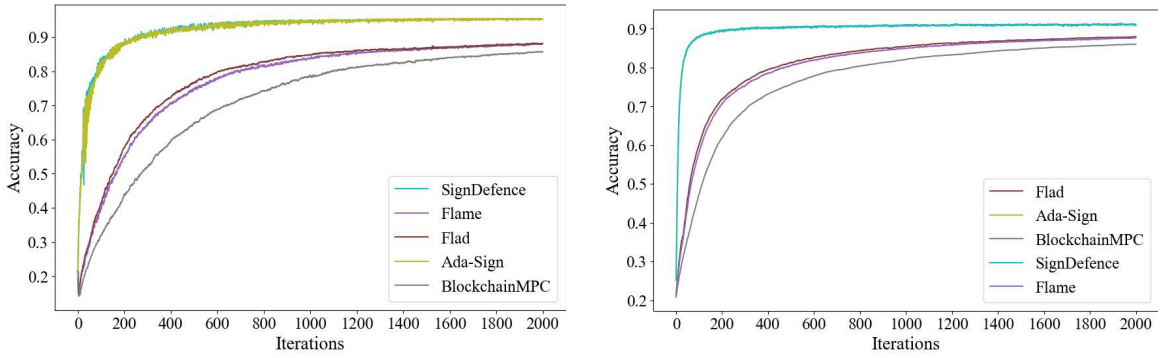
7.3.1 Performance Evaluation Under No Attack Scenario



(a) HAR (b) MNIST

Figure 7.2: No Attack: Evaluation of Accuracy for Established Techniques.

The experimental evaluation of *Ada-Sign* under benign conditions demonstrates its effectiveness in maintaining model accuracy while providing robust aggregation capabilities. As illustrated in Fig. 7.2, *Ada-Sign* achieves competitive performance across both HAR and MNIST datasets when



(a) HAR (b) MNIST

Figure 7.3: No Attack: Evaluation of Accuracy for Recent Techniques.

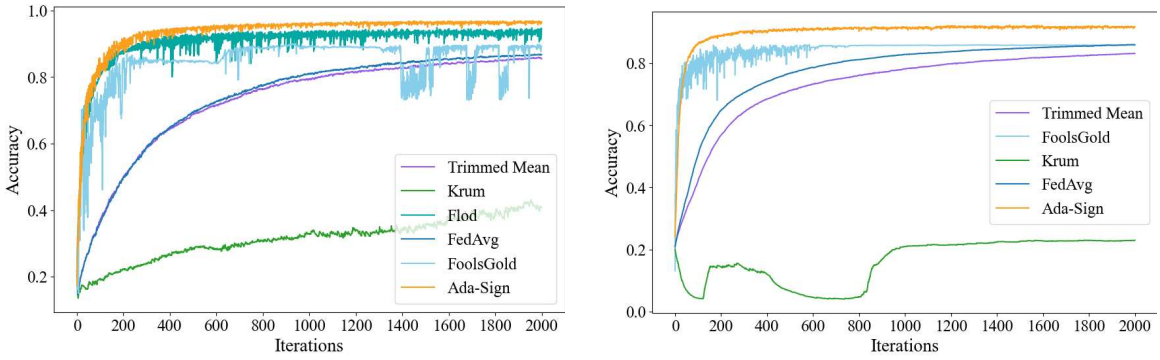
compared to established FL techniques.

In the HAR dataset evaluation as shown in Fig. 7.2a, *Ada-Sign* demonstrates accuracy improvements of 7.28% over Trimmed-Mean, 10.23% over Krum, and 22.83% over FoolsGold, while maintaining comparable performance to FedAvg with about 6.83% improvement. *Ada-Sign*'s results are close to Flod's, with only a 0.90% accuracy improvement, indicating Flod's sign gradient aligns with *Ada-Sign* in the absence of attacks. For the MNIST dataset evaluation as shown in Fig. 7.2b, *Ada-Sign* maintains consistent performance advantages, achieving 5.58% improvement over Trimmed-Mean and 6.23% over Krum. Notably, FoolsGold demonstrates a 5.84% degradation, suggesting *Ada-Sign*'s utility in certain benign scenarios. Flod was unstable during the MNIST evaluation setup, leading to complete training failure, and was therefore removed from the results.

When compared to recent techniques as shown in Fig. 7.3, *Ada-Sign* demonstrates superior performance with improvements of 0.15% over *SignDefence* and 3.67% over Flame, while maintaining competitive results against the BlockchainMPC approach with a 5.48% improvement in MNIST evaluation. On the HAR dataset, *SignDefence* shows similar degradation of 0.08%, while Flame and Flad exhibit substantial performance gaps of 7.36% and 7.49% respectively. BlockchainMPC demonstrates the most significant performance deficit of 10.02%.

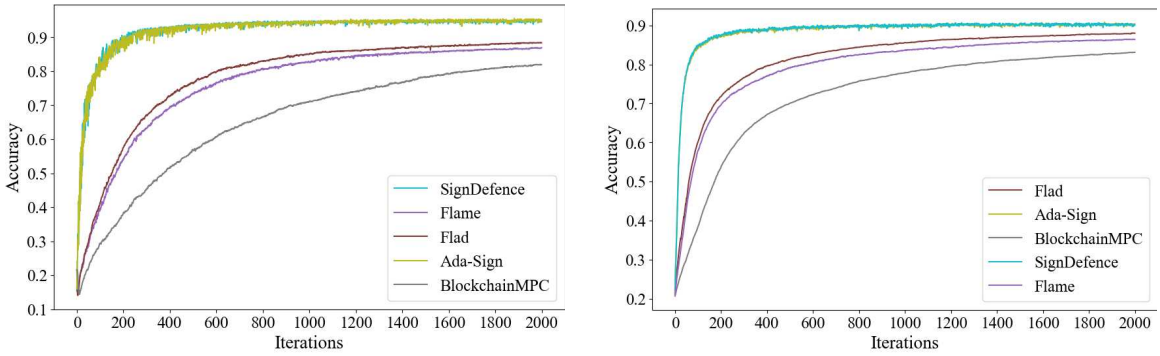
7.3.2 Resilience Against Krum Attack

The evaluation under Krum attack scenarios reveals *Ada-Sign*'s superior defensive capabilities against sophisticated aggregation-based attacks. As demonstrated in Fig. 7.4, *Ada-Sign* significantly



(a) HAR (b) MNIST

Figure 7.4: Attack on Krum: Evaluation of Accuracy for Established Techniques.



(a) HAR (b) MNIST

Figure 7.5: Attack on Krum: Evaluation of Accuracy for Recent Techniques.

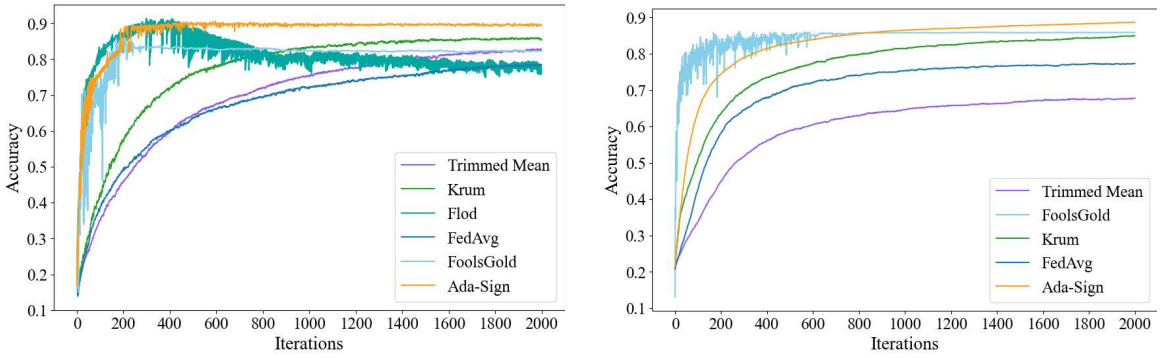
outperforms existing techniques in both datasets. The analysis of the obtained results is presented below.

In the HAR dataset, *Ada-Sign* achieves remarkable improvements of 57.25% over the targeted Krum technique, 10.11% over Trimmed-Mean,6.24% over FoolsGold, and 8.85% over FedAvg. However, *Ada-Sign* shows only 0.5% improvement over Flod, which can be attributed to the sign-based similar aggregation strategy. The vulnerability of Krum to its own attack variant is clearly evident, while *Ada-Sign*'s dynamic thresholding mechanism effectively mitigates the attack's impact. In the MNIST dataset, similar trends are observed with *Ada-Sign* demonstrating 74.55% improvement over Krum, 7.89% over Trimmed-Mean, 4.85% over FoolsGold, and 4.76% over FedAvg.

When evaluated against recent techniques in Fig. 7.5, *Ada-Sign* maintains its superiority with improvements of 0.22% over *SignDefence*, 4.29% over Flame, 2.51% over Flad, and 7.92% over BlockchainMPC approaches, demonstrating its robustness across different defensive paradigms in

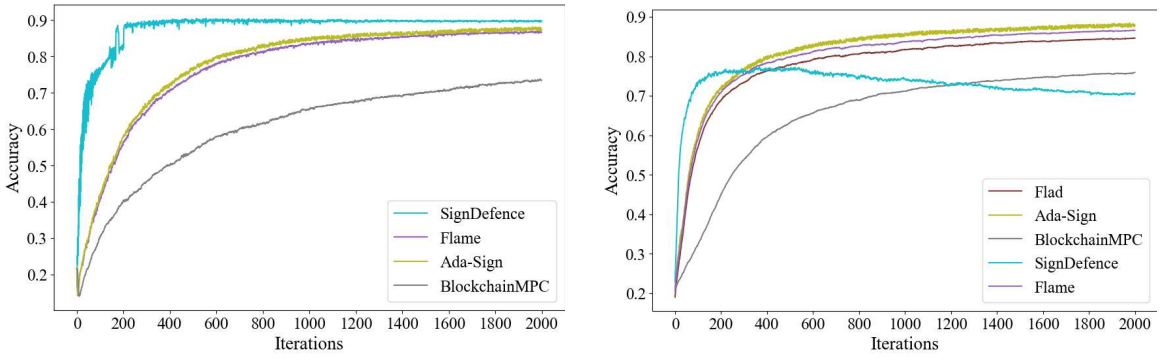
MNIST evaluation. While in the HAR dataset evaluation, it aligns with *SignDefence* with only 0.70% improvement over it because of similar gradient sign-based aggregation. However, Flame and Flad show substantial vulnerabilities due to degradation of 8.60% and 7.08%, respectively. BlockchainMPC exhibits the highest susceptibility with 13.88% performance loss in HAR dataset evaluation.

7.3.3 Defence Against Trimmed-Mean Attack



(a) HAR (b) MNIST

Figure 7.6: Attack on Trimmed-Mean: Evaluation of Accuracy for Established Techniques.



(a) HAR (b) MNIST

Figure 7.7: Attack on Trimmed-Mean: Evaluation of Accuracy for Recent Techniques.

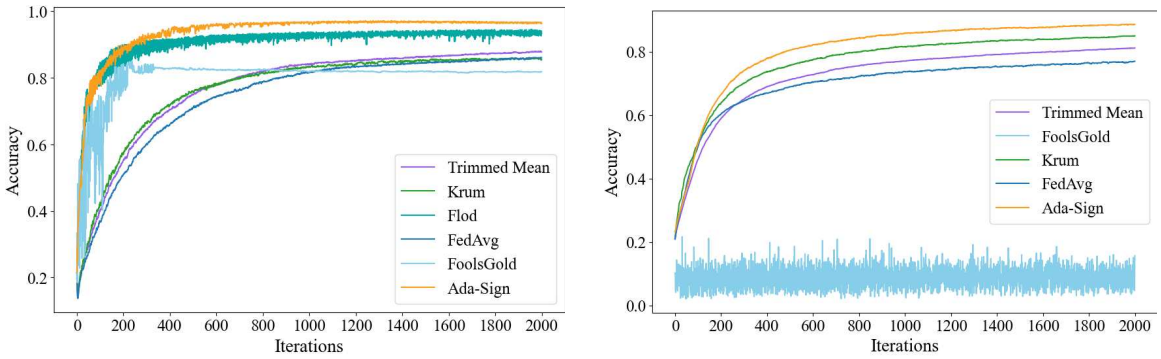
Ada-Sign's performance under Trimmed-Mean attack scenarios showcases its adaptability to more complex attack vectors targeting robust aggregation mechanisms. The results presented in Fig. 7.6 indicate *Ada-Sign*'s effectiveness in maintaining model integrity when adversaries specifically target distance-based aggregation methods. The analysis of the obtained results is presented

below.

In the HAR dataset, *Ada-Sign* achieves 5.31% improvement over the targeted Trimmed-Mean technique, 2.01% over Krum, 5.71% over FoolsGold, and 10.09% over FedAvg. The Flod technique demonstrates the worst performance with 12.90% accuracy loss over the proposed *Ada-Sign*, suggesting the effectiveness of sign-based techniques. For the MNIST dataset, *Ada-Sign* maintains consistent defensive capabilities with 5.19% improvement over Trimmed-Mean, 24.31% over Krum, 25.10% over FoolsGold, and 16.89% FedAvg, indicating its resilience against non-targeted attacks.

Comparison with recent techniques in Fig. 7.7 shows *Ada-Sign*'s competitive performance against *SignDefence*. In the HAR dataset, *Ada-Sign* experiences a 2.9% loss in accuracy compared to *SignDefence* due to the introduction of DP. However, in MNIST evaluation, *SignDefence* could not sustain due to the non-adaptive threshold mechanism and therefore, *SignDefence* loses almost 34% accuracy to *Ada-Sign*. In both datasets, Flame loses almost 25% accuracy to *Ada-Sign*. Blockchain-MPC, on the other hand, remains close to *Ada-Sign* in HAR dataset evaluation. However, it experiences 15.32% degradation in the MNIST dataset, demonstrating the dataset's impact.

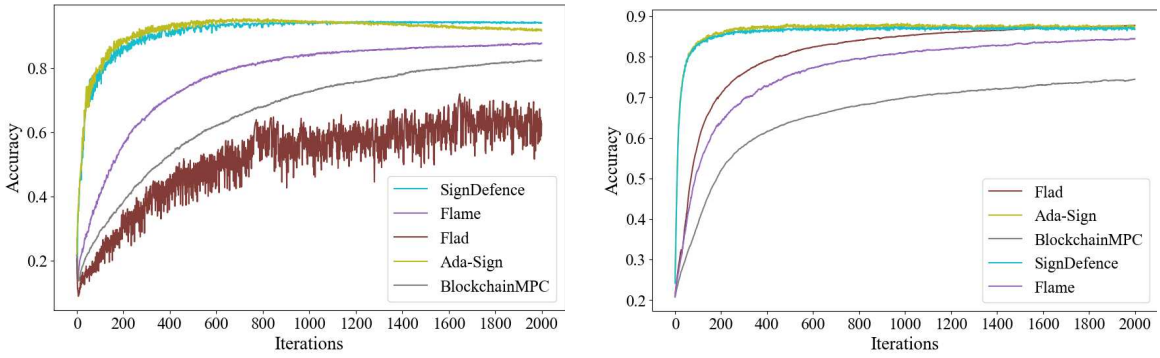
7.3.4 Mitigation of Label Flipping Attacks



(a) HAR (b) MNIST

Figure 7.8: Label Flipping Attack: Evaluation of Accuracy for Established Techniques.

The evaluation under Label Flipping attacks demonstrates *Ada-Sign*'s capability to detect and mitigate data poisoning attempts that target model training integrity. As shown in Fig. 7.8, *Ada-Sign* exhibits varying performance depending on the dataset characteristics and attack sophistication. The



(a) HAR (b) MNIST

Figure 7.9: Label Flipping Attack: Evaluation of Accuracy for Recent Techniques.

analysis of the obtained results is presented below.

In the HAR dataset, *Ada-Sign* achieves modest improvements of 1.57% over Trimmed-Mean, 4.4% over Flod, 3.49% over FedAvg, and 4.11% over Krum, while FoolsGold demonstrates bad performance with 8.21% degradation of accuracy compared to *Ada-Sign*. In the MNIST dataset, *Ada-Sign* maintains defensive capabilities with 3.78% improvement over Trimmed-Mean, while FoolsGold demonstrates the worst performance of 81.34% degradation of accuracy compared to *Ada-Sign*, suggesting its particular weakness against label manipulation attacks in image classification tasks. Krum is the close contender having only 0.72% less accuracy than the proposed *Ada-Sign* technique. However, FedAvg shows 8.77% degraded accuracy compared to *Ada-Sign*, showing consistently bad performance across MNIST datasets.

When compared to recent techniques in Fig. 7.9, *Ada-Sign* demonstrates superior performance. Specifically, on the HAR dataset, Flame demonstrates exceptional resilience with only 1.57% accuracy degradation, while *SignDefence* shows moderate vulnerability at 5.26%. Flad suffers the most severe impact with 29.58% performance loss, and BlockchainMPC exhibits 7.51% accuracy reduction. In contrast, *SignDefence* exhibits a 2.76% accuracy difference from *Ada-Sign*, and Flame shows a 0.12% difference in MNIST evaluation. Also, the BlockchainMPC approach suffers a substantial 11.74% accuracy loss relative to *Ada-Sign*, further highlighting the diverse defensive capabilities required for various attack scenarios.

7.4 Chapter Summary

This chapter presented a novel DP-enabled sign-based adaptive defence mechanism named *Ada-Sign*, which incorporates an adaptive threshold to empower sign-based aggregation. The comprehensive evaluation reveals that *Ada-Sign*'s dynamic thresholding mechanism and leaky weighted aggregation provide robust defence against multiple attack vectors while maintaining competitive performance in attack and non-attack scenarios. The proposed technique's ability to automatically adjust weights based on Jaccard similarity and MAD computation eliminates manual threshold tuning. The incorporation of DP during gradient exchange provides additional protection against inference and reconstruction attacks, enhancing its overall security posture. The experimental results demonstrate that no single defensive technique achieves optimal performance across all attack scenarios and datasets. However, *Ada-Sign* consistently demonstrates performance improvements, typically reaching 3-20% accuracy enhancements over existing methods in most scenarios. However, the improvement range varies significantly depending on the attack type and dataset characteristics, with some techniques showing minimal differences as low as 0.% in some instances. In comparison, others exhibit substantial performance gaps ranging from 50% to 80% under specific attack conditions. *Ada-Sign*'s strength lies in its consistent performance across diverse attack vectors, making it suitable for environments where attack types may vary or be unknown. The technique's superior performance against Krum attacks, Trimmed-Mean attacks, Flame variants and blockchain-based approaches indicates that *Ada-Sign* provides a practical balance between security, performance, and computational efficiency in FL deployments.

Chapter 8

SignMPC: Privacy-Preserving Federated Learning with Sign-Based Secure Aggregation

Despite its inherent privacy-by-design advantages, conventional FL architectures face two fundamental and often orthogonal challenges: privacy preservation and robustness against adversarial attacks [131]. Firstly, even without direct exposure to raw data, the shared model updates can inadvertently leak sensitive information about client datasets through various inference attacks [74]. This necessitates the integration of rigorous privacy-enhancing technologies, such as DP or SMPC [90]. Secondly, FL systems are highly vulnerable to malicious clients, commonly referred to as Byzantine attackers [64], who can inject poisoned or divergent model updates. Such attacks can significantly degrade the global model's performance, compromise its integrity, or even introduce a backdoor, rendering the collaborative learning process unreliable. Addressing these challenges in isolation often leads to suboptimal solutions; for instance, strong DP guarantees can impede model utility [96], while many robust aggregation methods lack integrated privacy assurances. The development of robust and privacy-preserving FL frameworks is therefore a critical research imperative. Existing approaches often specialize in either privacy or robustness, rarely providing a comprehensive and integrated solution that effectively balances the intricate trade-offs between privacy guarantees,

Byzantine resilience, and computational efficiency. The need for a unified framework that can simultaneously safeguard data confidentiality and ensure model integrity under adversarial conditions is becoming increasingly pressing as FL systems scale and deploy in real-world applications.

This chapter introduces *SignMPC*, a novel FL aggregation framework to provide simultaneous robustness against inference attacks and strong privacy guarantees for client data. *SignMPC* employs a multi-layered defence strategy, integrating state-of-the-art privacy-enhancing techniques with an innovative, robust aggregation mechanism. The framework's core novelty lies in its ability to adaptively aggregate client updates based on directional consensus, even when those updates have been perturbed for privacy and transmitted securely. The key contributions of this work are summarized as follows:

- A unified, multi-layered FL aggregation framework, *SignMPC*, that synergistically combines DP, SMPC, and a novel robust aggregation strategy to offer comprehensive protection against both privacy breaches and Byzantine attacks.
- An optimized robust aggregation algorithm which dynamically weights client contributions based on an adaptive Jaccard similarity threshold of their gradient signs. This approach offers superior resilience to data poisoning and model divergence attacks compared to traditional methods.
- The seamless integration of client-side DP to perturb individual gradient updates, ensuring privacy guarantees for client data before transmission.
- The incorporation of SMPC principles to facilitate the secure transmission and aggregation of client updates, preventing the central server from observing raw, individual gradient contributions even after DP perturbation.

8.1 Design

This section presents the design and architecture of the proposed *SignMPC* technique.

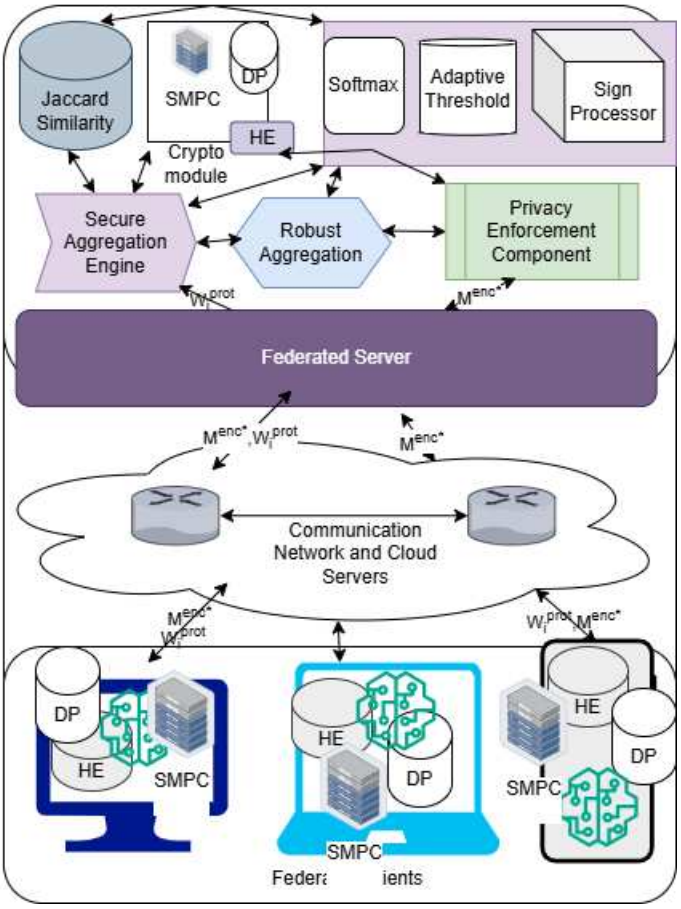


Figure 8.1: The Architecture of SignMPC.

8.1.1 SignMPC Architecture

Fig. 8.1 presents *SignMPC* architecture which consists of three core layers working in concert to enable privacy-preserving FL. At the client layer, distributed edge devices perform local model training while applying multiple privacy safeguards. Each client independently computes model updates, adds DP noise to obscure individual data contributions, and applies SMPC masks to enable confidential aggregation. The communication layer establishes secure channels between clients and servers, handling both encrypted model transmissions and secure mask exchanges required for the SMPC protocol while protecting participant metadata. The server layer coordinates the FL process through several specialized modules: a secure aggregation engine that combines encrypted client

updates while preserving confidentiality, a robust aggregation module that computes weighted averages using adaptive client scoring, and a privacy enforcement component that applies additional server-side DP. The architecture integrates four key functional modules, including sign-based gradient processing for efficient compression, Jaccard similarity analysis for Byzantine robustness, softmax weighting for fair contribution assessment, and adaptive thresholding to adjust aggregation parameters based on observed client behaviour dynamically. Cryptographic primitives, including HE, DP, and SMPC, are strategically deployed across these layers to provide end-to-end protection while maintaining model utility. The system design enables efficient federated training across heterogeneous devices while defending against both privacy attacks and malicious participants through its layered security approach.

8.1.2 Client-Side Secure Training

Algorithm 10 Client-Side Secure Training

Input:
$\mathcal{D}_i$ ▷ Local dataset
$\mathcal{M}^{\text{prot}}$ ▷ Protected global model (HE-encrypted or plaintext)
B, ε, δ ▷ Minibatch size and DP parameters
Output:
$\mathbf{w}_i^{\text{prot}}$ ▷ Protected update (HE/SMPC/DP processed)

if HE enabled then
  $\mathcal{M} \leftarrow \text{HE}_{\text{Decrypt}}(\mathcal{M}^{\text{prot}})$
else
  $\mathcal{M} \leftarrow \mathcal{M}^{\text{prot}}$
end if
$\mathbf{w}_i \leftarrow \text{MinibatchSGD}(\mathcal{M}, \mathcal{D}_i, B)$
if DP enabled then
  $\mathbf{w}_i \leftarrow \mathbf{w}_i + \mathcal{N}(0, \sigma^2_{\text{client}})$
end if
if SMPC enabled then
  $\mathbf{w}_i \leftarrow \mathbf{w}_i + \text{SMPC}_{\text{Mask}}()$ ▷ Add secure mask
end if
if HE enabled then
  $\mathbf{w}_i^{\text{prot}} \leftarrow \text{HE}_{\text{Encrypt}}(\mathbf{w}_i)$
else
  $\mathbf{w}_i^{\text{prot}} \leftarrow \mathbf{w}_i$
end if
return $\mathbf{w}_i^{\text{prot}}$ ▷ Final protected update

Algorithm 10 presents the secure client-side training where the client-side protocol enforces privacy through a multi-layered approach, beginning with cryptographic transformations and the strategic addition of noise, ensuring that sensitive local data remains protected throughout the FL

process. The initial step for a client involves decrypting the global model parameters received from the server, which is a critical operation when HE is active. This decryption makes the model parameters accessible in plaintext for local computations, allowing the client's local training process to interact directly with the model's numerical values, as expressed by:

$$\mathcal{M} = \mathrm{HE}_{\mathrm{Decrypt}}(\mathcal{M}^{\mathrm{enc}}) \tag{28}$$

Should HE not be employed, the client directly utilizes the global model parameters in their plaintext form. Following this, each client proceeds with local training using its private dataset $\mathcal{D}_i$. This training predominantly involves applying minibatch stochastic gradient descent (SGD) with a predefined learning rate η and a specified batch size B. The objective is to compute parameter updates that effectively minimize the local loss function, $\mathcal{L}$, over a sampled minibatch $\mathcal{B}_t$. The resulting local update $\Delta\mathbf{w}_i$ is directly derived from the negative gradient of the loss function with respect to the current global model parameters:

$$\Delta\mathbf{w}_i = -\eta\nabla\mathcal{L}(\mathcal{M}, \mathcal{B}_t) \tag{29}$$

This $\Delta\mathbf{w}_i$ intrinsically represents the client's proposed modification to the global model, reflecting the insights gained from its unique local data. Before this update is transmitted back to the central server, a series of privacy-preserving mechanisms are sequentially applied to construct the protected update $\mathbf{w}_i^{\mathrm{prot}}$. These mechanisms are designed to obscure the raw, individual contributions while maintaining the aggregate's utility. They typically include the addition of carefully calibrated noise for DP and the application of cryptographic masks for SMPC. Specifically, the client's local model parameters after training, denoted as $\mathbf{w}_i$ (which are $\mathcal{M} + \Delta\mathbf{w}_i$), are perturbed by additive Gaussian noise sampled from a zero-mean distribution with variance σ_c^2, specifically for client-side DP. Concurrently, an SMPC mask, constructed from pairwise random shares exchanged securely between clients, is added. This mask, $\sum_{j\neq i}(\mathbf{m}_{i,j} - \mathbf{m}_{j,i})$, ensures that individual contributions are cryptographically obscured from the server, yet can be algebraically cancelled during aggregation. The comprehensive formulation for the protected update, encapsulating all these layers, is given by:

$$\mathbf{w}_i^{\text{prot}} = \text{HE}_{\text{Encrypt}} \left(\mathbf{w}_i + \underbrace{\mathcal{N}(0, \sigma_c^2)}_{\text{DP noise}} + \underbrace{\sum_{j \neq i} (\mathbf{m}_{i,j} - \mathbf{m}_{j,i})}_{\text{SMPC mask}} \right) \tag{30}$$

The entire sum is then subjected to homomorphic encryption if HE is enabled. This multi-faceted protection scheme guarantees strong confidentiality and unlinkability, meaning that the server or other colluding parties cannot infer individual clients' data and their specific contributions. Despite these transformations, the mathematical structure required for accurate and secure aggregation on the server side is meticulously preserved. The resulting encrypted output $\mathbf{w}_i^{\text{prot}}$ effectively encodes the directional information of the updates while robustly preventing the disclosure of sensitive individual training data characteristics or exact client contributions.

8.1.3 Server-Side SignMPC Aggregation

Algorithm 11 presents a server-side *SignMPC* aggregation protocol, meticulously engineered to process protected client updates securely and robustly, culminating in an updated global model that benefits from the collective intelligence of the distributed network while mitigating adversarial influences. The process commences by reversing the cryptographic transformations applied on the client side to unveil the underlying updates in a controlled, privacy-preserving manner. Specifically, if HE was utilized, the received encrypted protected updates $\mathbf{w}_i^{\text{prot}}$ are first homomorphically decrypted. Following this, if secure multi-party computation (SMPC) was active, the cryptographic masks applied by clients are algebraically removed, allowing the server to work with the sum of the true updates without seeing individual components. This initial transformation sequence can be conceptually represented as:

$$\mathbf{w}_i = \text{SMPC}_{\text{Reveal}} \left(\text{HE}_{\text{Decrypt}}(\mathbf{w}_i^{\text{prot}}) \right) \tag{31}$$

Once the effective client updates $\mathbf{w}_i$ are recovered, the server proceeds to compute the individual gradients $\mathbf{g}_i$ that correspond to these updates. This is typically achieved by calculating the difference between the current global model parameters $\mathcal{M}$ and the client's transmitted update $\mathbf{w}_i$. To ensure

Algorithm 11 Server-Side SignMPC Aggregation

Input:
$\{\mathbf{w}_1^{\text{prot}}, \ldots, \mathbf{w}_N^{\text{prot}}\}$ ▷ Protected client updates (HE-encrypted or SMPC-masked)
$\mathcal{M}^{\text{enc}}$ ▷ Encrypted global model parameters
$\tau_{\text{base}}, \alpha, \beta, \eta$ ▷ Threshold parameters
ε, δ ▷ DP privacy parameters
Output:
$\mathcal{M}^{\text{enc}}{}_*$ ▷ Updated encrypted global model
if HE enabled **then**
  $\{\mathbf{w}_i\} \leftarrow \text{HE}_{\text{Decrypt}}(\{\mathbf{w}_i^{\text{prot}}\})$
  $\mathcal{M} \leftarrow \text{HE}_{\text{Decrypt}}(\mathcal{M}^{\text{enc}})$
else
  $\{\mathbf{w}_i\} \leftarrow \{\mathbf{w}_i^{\text{prot}}\}$ ▷ Assume plaintext if no HE
end if
if SMPC enabled **then**
  $\{\mathbf{w}_i\} \leftarrow \text{SMPC}_{\text{Reveal}}(\{\mathbf{w}_i\})$ ▷ Remove masks
end if
$\mathbf{g}_i \leftarrow \mathcal{M} - \mathbf{w}_i$ ▷ Compute gradient from update
$\mathbf{g}_i \leftarrow \text{clip}(\mathbf{g}_i, -\tau_{\text{clip}}, \tau_{\text{clip}})$
if DP enabled **then**
  $\mathbf{g}_i \leftarrow \mathbf{g}_i + \mathcal{N}(0, \sigma_{\text{server}}^2)$
end if
$\mathbf{G}_{\text{agg}} \leftarrow \text{mean}(\{\mathbf{g}_i\})$ ▷ Standard aggregation
$\mathbf{s}_i \leftarrow \text{sign}(\mathbf{g}_i)$
$\mathbf{s}_{\text{ref}} \leftarrow \text{sign}(\mathbf{G}_{\text{agg}})$ or consensus signs
$J_i \leftarrow \text{JaccardSimilarity}(\mathbf{s}_i, \mathbf{s}_{\text{ref}})$
$\tau \leftarrow \text{AdaptiveThreshold}(\{\mathbf{g}_i\}, \tau_{\text{base}}, \alpha, \beta)$
$w_i \leftarrow \text{softmax}(J_i - \tau)$
$\mathcal{M}^* \leftarrow \mathcal{M} - \eta \sum_{i=1}^{N} w_i \mathbf{s}_i$
$\mathcal{M}^{\text{enc}*} \leftarrow \text{HE}_{\text{Encrypt}}(\mathcal{M}^*)$ ▷ Re-encrypt if HE enabled
return $\mathcal{M}^{\text{enc}*}$

the stability of the aggregation process and prevent any single client's potentially large or malicious gradient from disproportionately influencing the global model, a robust clipping operation is applied. This operation bounds the L2 norm of each gradient, with $\|\cdot\|_2$ representing the Euclidean norm:

$$\mathbf{g}_i = \text{clip}\left(\frac{\mathcal{M} - \mathbf{w}_i}{\|\mathcal{M} - \mathbf{w}_i\|_2}, -\tau_{\text{clip}}, \tau_{\text{clip}}\right) \tag{32}$$

Here, τ_{clip} defines the symmetric clipping threshold. After this essential clipping step, if server-side DP is enabled, carefully calibrated Gaussian noise $\mathcal{N}(0, \sigma_s^2)$ is added to the gradients, further

enhancing privacy by obscuring the exact aggregate. The distinguishing characteristic of this aggregation strategy is its reliance on a sign-based approach, which inherently confers significant robustness against various types of Byzantine attacks, particularly those involving magnitude manipulation. The algorithm calculates the element-wise sign of each gradient, $\text{sign}(\mathbf{g}_i)$, which essentially captures the directional information of the update. These signed gradients are then combined using adaptively computed weights w_i. These weights are dynamically derived from Jaccard similarity scores and an adaptive threshold, the calculation of which is meticulously detailed in Algorithm 12. The global model is then updated by subtracting the learning rate η multiplied by the weighted sum of these signed gradients, coupled with the optional server-side DP noise addition:

$$\mathcal{M}^* = \mathcal{M} - \eta \left(\sum_{i=1}^{N} w_i \text{sign}(\mathbf{g}_i) + \mathcal{N}(0, \sigma_s^2) \right) \tag{33}$$

This formulation not only ensures DP at the server level by introducing noise to the aggregated update but also leverages the inherent resilience and convergence properties of sign-based gradient descent. The clipping operation on individual gradients fundamentally limits the maximum influence any single client can exert. At the same time, the additive noise provides formal privacy guarantees for the final aggregated result, preventing the server from reconstructing individual contributions. Finally, suppose HE was initially enabled for model parameters. In that case, the updated global model $\mathcal{M}^*$ is re-encrypted before being securely broadcast back to clients for the subsequent training round.

8.1.4 Adaptive Threshold Estimation

Algorithm 12 presents the adaptive threshold estimation. The adaptive threshold estimation mechanism is a pivotal component for dynamically adjusting client weights based on the intrinsic geometric similarity of their gradient contributions, thereby significantly enhancing the robustness of the overall aggregation process against adversarial clients. This sophisticated approach is designed to autonomously identify and appropriately down-weight malicious or outlying updates without requiring explicit prior knowledge or assumptions about the nature of the attack. The estimation process is initiated by establishing a robust and central reference direction for the collection

Algorithm 12 Adaptive Threshold Estimation

Input:
$\{\mathbf{g}_i\}_{i=1}^N$ ▷ Gradient set
τ_{base} ▷ Initial threshold
α ▷ Adaptation rate
β ▷ MAD scaling factor
Output:
τ^* ▷ Computed threshold
$\mathbf{m} \leftarrow \text{median}(\{\mathbf{g}_i\})$ ▷ Median gradient
$\mathbf{b}_i \leftarrow (\mathbf{g}_i > \mathbf{m})$ ▷ Binary encoding
$\mathbf{b}_{\text{ref}} \leftarrow \mathbf{b}_N$ ▷ Reference encoding
$J_i \leftarrow \frac{|\mathbf{b}_i \cap \mathbf{b}_{\text{ref}}|}{|\mathbf{b}_i \cup \mathbf{b}_{\text{ref}}|}$ ▷ Jaccard similarity
$\text{MAD} \leftarrow \beta \cdot \text{median}(\{|J_i - \text{median}(\{J_i\})|\})$ ▷ Scaled MAD
$\tau^* \leftarrow \tau_{\text{base}} - \alpha \cdot \text{MAD}$ ▷ Adjusted threshold
$\tau^* \leftarrow \max(0.1, \min(0.9, \tau^*))$ ▷ Clamped value
return τ^*

of client gradients. This is precisely achieved by computing the element-wise median of the entire set of client gradients $\{\mathbf{g}_i\}_{i=1}^N$:

$$\mathbf{m} = \text{median}(\{\mathbf{g}_i\}_{i=1}^N) \tag{34}$$

The median is deliberately chosen over the mean due to its superior resilience to extreme outliers, making it a more stable measure of central tendency in potentially adversarial environments. Subsequently, each client's raw gradient $\mathbf{g}_i$ is transformed into a binary vector by comparing its elements against the corresponding elements of this computed median. This creates a boolean vector $\mathbb{I}(\mathbf{g}_i > \mathbf{m})$, where an element is set to 1 if the corresponding component of $\mathbf{g}_i$ is greater than the median component, and zero otherwise. A specific binarized gradient, typically the last one received ($\mathbf{g}_N$), is designated as the reference binarized gradient $\mathbf{g}_{\text{ref}}$. Similarity scores are then meticulously computed for each client's binarized gradient against this reference using the Jaccard index. This metric quantifies the overlap of the directional agreement:

$$J_i = \frac{\langle \mathbb{I}(\mathbf{g}_i > \mathbf{m}), \mathbb{I}(\mathbf{g}_{\text{ref}} > \mathbf{m}) \rangle}{\|\mathbb{I}(\mathbf{g}_i > \mathbf{m}) \cup \mathbb{I}(\mathbf{g}_{\text{ref}} > \mathbf{m})\|_0} \tag{35}$$

In this formulation, $\langle \cdot, \cdot \rangle$ denotes the dot product, effectively representing the size of the intersection of active (1-valued) bits between the two binary vectors. The denominator, $\| \cdot \|_0$, represents the number of non-zero elements, which corresponds to the size of the union of active bits. This

Jaccard similarity score precisely quantifies the degree of agreement in the directional information conveyed by the binarized gradients. To render the threshold truly adaptive, these similarity metrics are combined with a robust estimate of the dispersion of the J_i scores, specifically the Median Absolute Deviation (MAD). The MAD is scaled by a constant factor (e.g., 1.4826, which is used to make it a consistent estimator for the standard deviation under Gaussian distribution assumptions) to provide a more stable and reliable measure of variability in the Jaccard scores. The final adaptive threshold τ^* is then judiciously computed by adjusting a predetermined base threshold τ_{base} based on this scaled MAD:

$$\tau^* = \text{clip}\left(\tau_{\text{base}} - \alpha \cdot \underbrace{1.4826 \cdot \text{median}(|J_i - \tilde{J}|)}_{\text{MAD estimate}}, 0.1, 0.9\right) \tag{36}$$

Here, $\tilde{J}$ signifies the median of the set of Jaccard similarity scores, and α is a configurable adaptation rate that controls the sensitivity of the threshold to variations in client similarity. The $\text{clip}(\cdot, 0.1, 0.9)$ function serves as a crucial safeguard, ensuring that the adaptive threshold remains within a pragmatic and practical operational range, typically bounded between 0.1 and 0.9. This dynamic adjustment mechanism enables the aggregation process to automatically adapt to the observed consensus among clients, effectively diminishing the influence of outlier updates while preserving the valuable contributions from honest participants, even in scenarios characterised by non-IID data distributions.

8.1.5 Secure Aggregation Protocol

Algorithm 13 Secure Aggregation (SMPC)

Input:
$\{\tilde{\mathbf{g}}_i\}_{i=1}^N$ ▷ Masked gradients
$\{\mathbf{m}_i\}_{i=1}^N$ ▷ Client masks
Output:
$\mathbf{G}_{\text{agg}}$ ▷ Aggregated result
$\mathbf{G}_{\text{masked}} \leftarrow \frac{1}{N}\sum_{i=1}^N \tilde{\mathbf{g}}_i$ ▷ Mean of masked gradients
$\mathbf{M}_{\text{avg}} \leftarrow \frac{1}{N}\sum_{i=1}^N \mathbf{m}_i$ ▷ Average mask
$\mathbf{G}_{\text{agg}} \leftarrow \mathbf{G}_{\text{masked}} - \mathbf{M}_{\text{avg}}$ ▷ Demasked result
return $\mathbf{G}_{\text{agg}}$

The Secure Aggregation Protocol, presented in Algorithm 13, is meticulously implemented based on the principles of Secure Multi-Party Computation (SMPC). It is specifically engineered to

aggregate client updates in a manner that robustly preserves the privacy and confidentiality of individual client contributions. This is achieved through an ingenious cryptographic masking scheme where clients add random, algebraically cancelling masks to their gradients before transmitting them to the central server. For each client i, its true, locally computed gradient $\mathbf{g}_i$ is transformed into a masked gradient $\tilde{\mathbf{g}}_i$ by incorporating a sum of pairwise random masks. These masks are securely exchanged between clients using established cryptographic protocols:

$$\tilde{\mathbf{g}}_i = \mathbf{g}_i + \sum_{j \neq i} (\mathbf{m}_{i,j} - \mathbf{m}_{j,i}) \tag{37}$$

In this formulation, $\mathbf{m}_{i,j}$ represents a randomly generated mask share that client i creates for client j, and $\mathbf{m}_{j,i}$ is the corresponding mask share client j creates for client i. The fundamental property of this construction is that for any given pair of clients (i, j), their respective mask shares $\mathbf{m}_{i,j}$ and $\mathbf{m}_{j,i}$ are designed to algebraically cancel each other out when summed collectively across all participating clients. The central server receives these masked gradients from all clients. Crucially, due to the linearity and the carefully designed cancellation property of these masks, when the server computes the sum or average of all received masked gradients, the sum of all individual masks precisely evaluates to zero. This allows the server to compute the exact aggregate of the true, unmasked gradients without ever gaining knowledge of any individual client's specific $\mathbf{g}_i$. The server computes the aggregated result $\mathbf{G}_{\text{agg}}$ as:

$$\mathbf{G}_{\text{agg}} = \frac{1}{N} \sum_{i=1}^{N} \tilde{\mathbf{g}}_i - \underbrace{\frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq i} (\mathbf{m}_{i,j} - \mathbf{m}_{j,i})}_{\text{Cancels out}} \tag{38}$$

The second term on the right-hand side, representing the sum of all pairwise masks across all clients, demonstrably evaluates to zero due to the symmetric nature of the mask generation ($\mathbf{m}_{i,j}$ and $\mathbf{m}_{j,i}$ effectively cancelling). This process effectively yields $\mathbf{G}_{\text{agg}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{g}_i$. This construction inherently provides information-theoretic privacy for the aggregation phase, implying that the security of individual contributions is maintained as long as a minimum number of clients (typically at least two in this pairwise mask scheme) remain honest and do not collude with the server or other malicious entities. The ingenious pairwise mask structure is fundamental to ensuring the correct cancellation

of the masks while rigorously preventing the server from learning any individual client's updates, thereby providing robust confidentiality guarantees for the entire aggregation process within the FL paradigm.

8.2 Experimental Setup

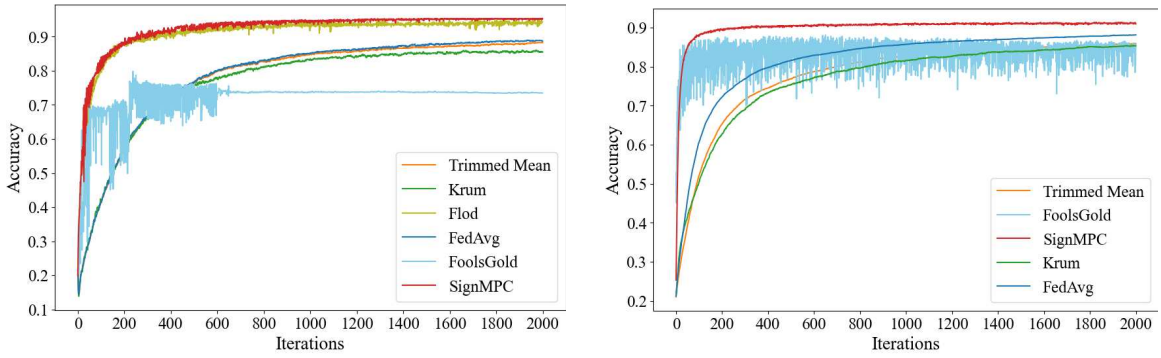
The *SignMPC* framework evaluation was conducted using comprehensive simulations implemented in PyTorch, building upon the SAFEFL platform [41]. All experiments were performed on Apple M1 Pro hardware with an 8-core CPU, 14-core GPU, 16GB unified memory, and 512GB SSD storage. The FL environment consisted of 30 clients and a central server, maintaining consistency with established evaluation protocols. The evaluation employed two datasets to demonstrate generalization across different data domains. The HAR [41] dataset comprised six activity classes collected from thirty participants with approximately 10,299 samples, while the MNIST [114] evaluation utilized the standard 10-digit classification task with 70,000 samples. Both datasets exhibited non-IID characteristics across clients to simulate real-world data heterogeneity. During this experiment, 20% of samples represent the test set and 5% of samples represent the validation set, where the rest is used for the training dataset.

The *SignMPC* framework incorporates three core privacy-preserving components with carefully calibrated parameters. The differential privacy component employs a privacy budget of $\varepsilon = 3.0$ and $\delta = 1e^{-4}$ with Gaussian noise calibration following $\sigma = \min(\sqrt{2\ln(1.25/\delta)} \times \text{sensitivity}/\varepsilon, 0.1)$ and gradient clipping bounded at L2 norm of 2.0. The secure multiparty computation component utilizes additive secret sharing with lightweight masking across all 30 clients, employing a threshold of $t = 10$ and maintaining a mask scale of 0.01 to minimize impact on gradient utility. The HE component operates with a 512-bit key size, utilizing a noise scale of $1e^{-4}$ and a modulus of $2^{31} - 1$ for computational efficiency. The robust aggregation algorithm employs a threshold base of $\tau_{\text{base}} = 0.7$ with an adaptive factor $\alpha = 0.05$, allowing the Jaccard similarity threshold to dynamically adjust between 0.1 and 0.9 based on client behavior patterns. Model training utilized a batch size of 64 with an initial learning rate of 0.01 across 2000 communication rounds. Attack simulation encompassed four distinct scenarios with 20% of clients (6 out of 30) configured as malicious, targeting 30% of

model parameters. The scenarios included no attack (baseline), Krum attack maximizing deviation using compromised parameters, Trimmed-Mean attack manipulating parameters toward extreme values, and Label Flipping attack altering class mappings following established Byzantine attack patterns. Comparative evaluation included Trimmed-Mean [143], Krum [15], Flod [32], FedAvg [79], BlockchainMPC [52], FoolsGold [40], Flad[114], *SignDefence*, and Flame [86] techniques. Performance evaluation focused exclusively on training accuracy as the primary metric, measured as the model's classification performance on the respective datasets.

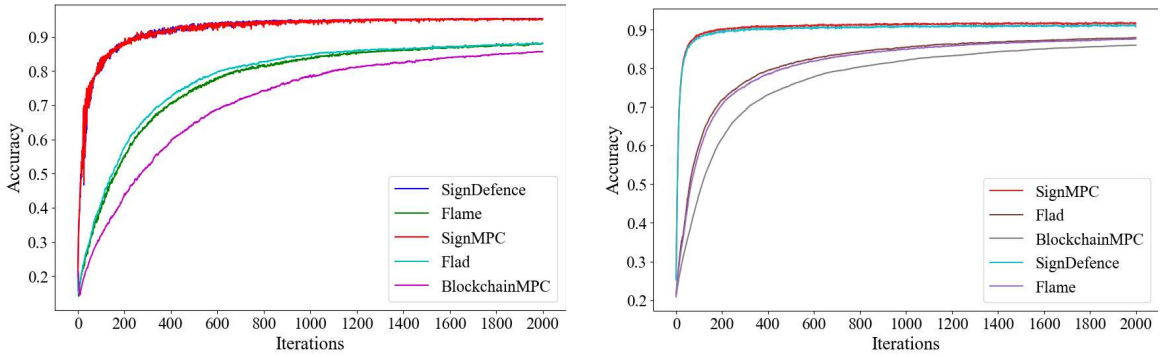
8.3 Results and Discussion

8.3.1 No Attack Scenario



(a) HAR (b) MNIST

Figure 8.2: No Attack Scenario: Evaluation of Accuracy for Established Techniques.



(a) HAR (b) MNIST

Figure 8.3: No Attack: Evaluation of Accuracy for Most Recent Techniques.

Figure 8.2 presents the performance evaluation of *SignMPC* against established defence mechanisms under benign conditions with no adversarial attacks present. *SignMPC* demonstrates optimal performance by maintaining the same accuracy as the baseline FL setup, indicating minimal overhead while preserving model utility. The comparative analysis reveals that existing defence mechanisms introduce varying degrees of performance degradation even in attack-free scenarios.

In the HAR dataset evaluation with established techniques, FoolsGold exhibits the highest accuracy degradation of 22.93%, followed by Krum at 10.34%, Trimmed-Mean at 7.40%, Flad at 7.60%, FedAvg at 6.95%, and Flod at 1.02% compared to *SignMPC*. Flod did not perform well in the MNIST evaluation and, therefore, was removed from the figure. However, FoolsGold is unstable during training and loses approximately 5.78% accuracy compared to the proposed *SignMPC* technique. Furthermore, the proposed method achieves 5.53% accuracy improvement over Trimmed-Mean, 6.18% over Krum, and 3.09% over FedAvg techniques.

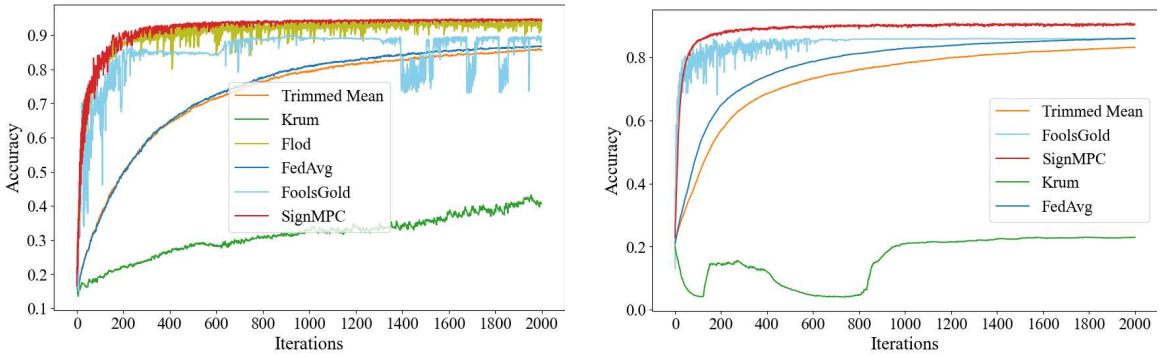
Figure 8.3 demonstrates the comparison with most recent techniques on the MNIST dataset, where Flame shows 7.48% degradation, Flad 7.60%, BlockchainMPC 10.13%, and *SignDefence* maintains close performance at 0.04% with *SignMPC*.

The results indicate that *SignMPC*'s design philosophy of preserving model utility during normal operations is successfully validated, while other defensive mechanisms impose computational or accuracy penalties even without adversarial presence. This characteristic is particularly significant for FL deployments where maintaining model performance is crucial for client adoption and system sustainability. The superior performance of *SignMPC* can be attributed to its robust aggregation algorithm, which dynamically weights client contributions based on an adaptive Jaccard similarity threshold of their gradient signs. This approach offers superior resilience to data poisoning and model divergence attacks compared to traditional methods, representing a key strength of *SignMPC*.

Notably, the integration of DP mechanisms, HE, and SMPC protocols within *SignMPC* does not degrade performance due to the adaptive thresholding mechanism. This is a significant advantage over conventional defence approaches that typically suffer from the privacy-utility trade-off, where enhanced privacy guarantees come at the cost of reduced model accuracy. The adaptive nature of *SignMPC*'s aggregation process ensures that privacy-preserving mechanisms complement rather

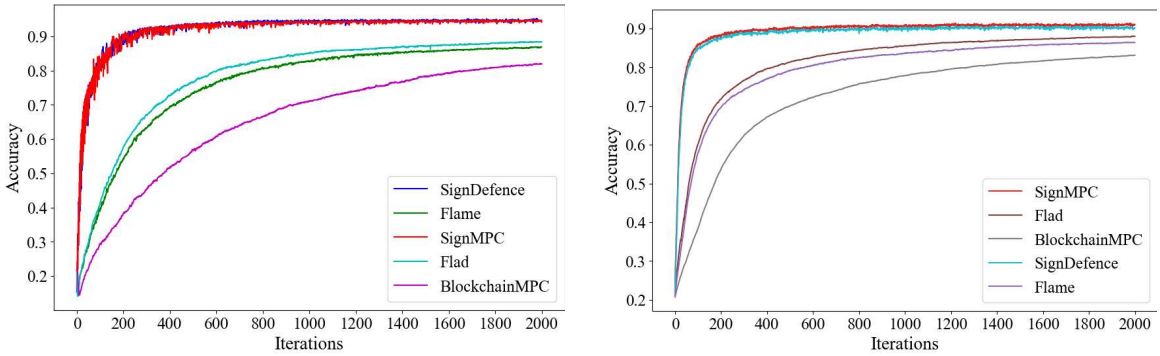
than compromise the learning objective.

8.3.2 Krum Attack Resilience



(a) HAR (b) MNIST

Figure 8.4: Attack on Krum: Evaluation of Accuracy for Established Techniques.



(a) HAR (b) MNIST

Figure 8.5: Attack on Krum: Evaluation of Accuracy for Most Recent Techniques.

Figure 8.4 illustrates the evaluation under Krum attack conditions for established techniques, revealing substantial performance variations across different defensive mechanisms. *SignMPC* maintains similar accuracy to the baseline FL setup, demonstrating robust resistance to this attack vector. In contrast, Krum itself suffers severe degradation, with accuracy reductions of 56.79% and 74.59% in HAR dataset evaluations, highlighting the vulnerability of the aggregation mechanism to targeted attacks that exploit its distance-based selection criteria.

Established defence techniques show mixed effectiveness, with Trimmed-Mean experiencing

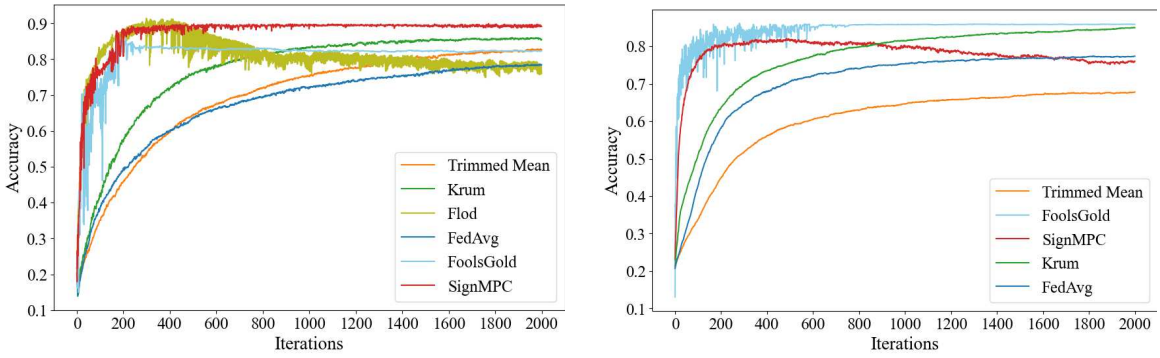
9.15% and 8.02% accuracy degradation, FedAvg recording 7.86% and 4.90%, and FoolsGold demonstrating 5.22% and 5.03% performance losses across the respective datasets. The proposed technique achieves remarkable improvements in MNIST evaluation, with 8.02% over Trimmed-Mean, 5.03% over FoolsGold, and a dramatic 74.59% accuracy enhancement over the targeted Krum method itself. Additionally, *SignMPC* demonstrates 4.90% over FedAvg, showcasing its robust defence capabilities against sophisticated Byzantine attacks that specifically target geometric median-based aggregation schemes.

Figure 8.5 shows the performance of the most recent defence mechanisms, which exhibit varying performance levels, with Flad showing 6.08% and 2.66% degradation, while Flame variants achieve 7.62% and 4.43% accuracy reductions, respectively. *SignDefence* demonstrates moderate resilience with performance degradation within acceptable ranges, while Blockchain-based approaches show 12.96% performance loss in the HAR dataset evaluation, maintaining 8.05% degradation in the MNIST dataset.

The superior performance of *SignMPC* under Krum attack conditions can be attributed to its sign-based gradient aggregation mechanism, which inherently resists distance-based manipulation strategies that specifically target Krum's aggregation logic. The robust aggregation algorithm dynamically weights client contributions based on an adaptive Jaccard similarity threshold of their gradient signs, offering superior resilience to data poisoning and model divergence attacks compared to traditional methods. This resistance mechanism ensures that malicious clients cannot effectively skew the aggregation process through carefully crafted model updates designed to exploit Euclidean distance calculations. Furthermore, the adaptive thresholding mechanism enables *SignMPC* to maintain consistent performance even when enhanced with DP, HE, and SMPC protocols, eliminating the typical privacy-utility trade-off observed in conventional approaches.

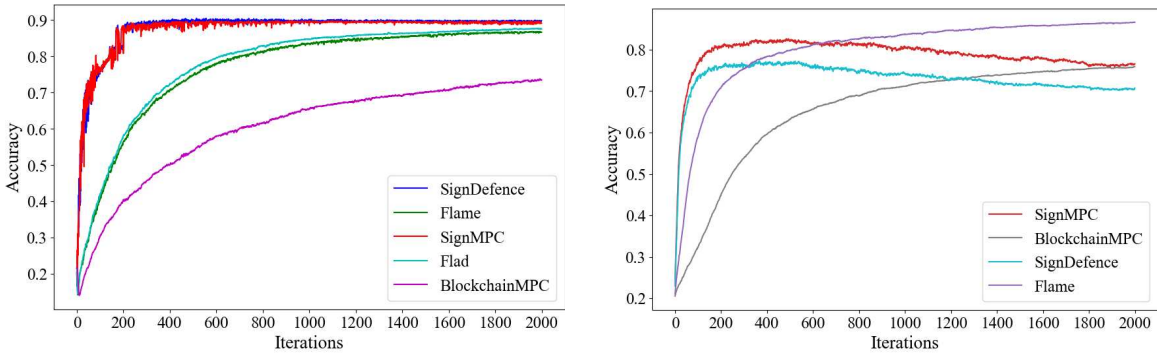
8.3.3 Trimmed-Mean Attack Analysis

Figure 8.6 presents the comparative evaluation under Trimmed-Mean attack scenarios among established techniques. The attack targets explicitly the median calculation mechanism inherent in Trimmed-Mean aggregation, where malicious clients craft updates that fall within the acceptable range but systematically bias the aggregation outcome.



(a) HAR (b) MNIST

Figure 8.6: Attack on Trimmed-Mean: Evaluation of Accuracy for Established Techniques.



(a) HAR (b) MNIST

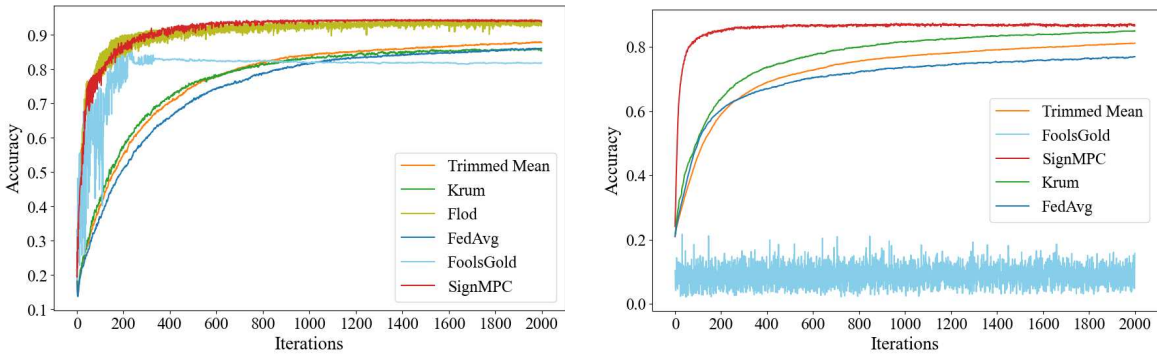
Figure 8.7: Attack on Trimmed-Mean: Evaluation of Accuracy for Most Recent Techniques.

Established defence approaches show significant performance degradation, with Trimmed-Mean itself experiencing a 7.30% accuracy reduction in the HAR dataset. Krum demonstrates 4.06% performance losses, while FedAvg shows 11.98% degradation in the same dataset. Among the established defences, Flod exhibits the highest vulnerability with 14.73% accuracy reduction in the HAR dataset, while FoolsGold maintains relatively stable performance with 7.69% in the HAR dataset. The weakness of the proposed *SignMPC* was revealed in the MNIST dataset when it was attacked using the Trimmed-Mean. The proposed technique loses 10.56%, 11.50%, and 1.80% accuracy to Krum, FoolsGold, and FedAvg, respectively. However, the Trimmed-Mean technique loses approximately 10.74% accuracy compared to the proposed *SignMPC* technique. These exceptional results can be attributed to the nature of the attack and the failure of the MAD-based adaptive threshold estimation algorithm in *SignMPC*, which requires further improvement.

Figure 8.7 demonstrates that the most recent defence mechanisms show improved resilience

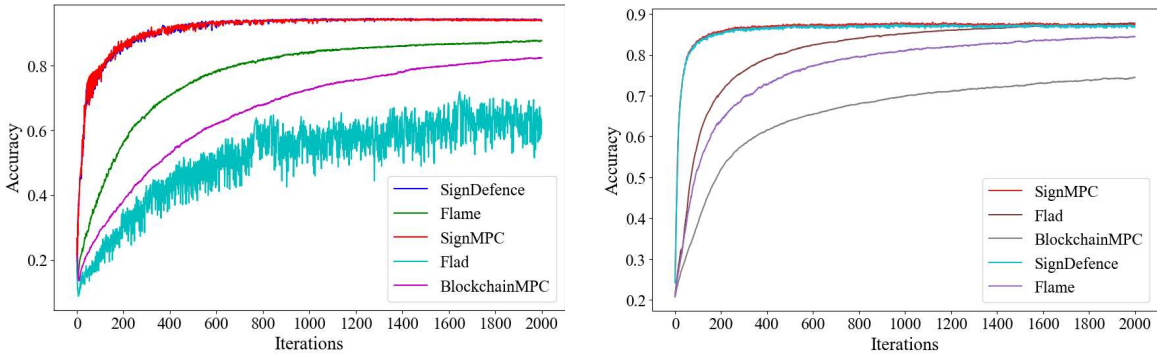
compared to established approaches, with Flad showing 1.75% accuracy loss, Flame variants achieving 2.80% degradation, and BlockchainMPC defence experiencing a 17.57% reduction in the HAR dataset evaluation. Similarly to the established approach, a comparison of both *SignDefence* and *SignMPC* reveals degraded accuracy in the MNIST evaluation. *SignMPC* loses 13.54% accuracy to Flad and 12.28% to Flame. However, *SignMPC* shows 3.6% better accuracy than *SignDefence* and achieves similar accuracy to BlockchainMPC.

8.3.4 Label Flipping Attack Evaluation



(a) HAR (b) MNIST

Figure 8.8: Label Flipping Attack: Evaluation of Accuracy for Established Techniques.



(a) HAR (b) MNIST

Figure 8.9: Label Flipping Attack: Evaluation of Accuracy for Most Recent Techniques.

Figure 8.8 shows the Label Flipping attack evaluation for established techniques, which presents

one of the most challenging scenarios for FL defence mechanisms, as it directly targets the fundamental learning objective through systematic label manipulation. *SignMPC* demonstrates exceptional resilience by maintaining consistent accuracy levels comparable to the baseline setup. This attack vector represents a particularly insidious threat where malicious clients systematically replace correct labels with incorrect ones, thereby degrading the global model's classification performance.

The comparative analysis reveals significant performance variations across different defence mechanisms when subjected to Label Flipping attacks. In the HAR dataset evaluation, FoolsGold experiences substantial degradation of 12.75%, while established approaches show moderate vulnerability with Krum at 8.85%, FedAvg at 8.26%, and Trimmed-Mean at 6.44%. Flod demonstrates exceptional resilience with minimal 0.50% accuracy reduction, while Flad shows significant vulnerability with 33.06% performance degradation.

Figure 8.9 reveals even more pronounced performance differences in the MNIST dataset evaluation, with FoolsGold experiencing severe degradation of 81.82%, indicating fundamental limitations in handling systematic label manipulation attacks. Other most recent defence mechanisms demonstrate varying degrees of resilience, with Flad showing 1.32% degradation, Krum at 1.84%, Flame variants achieving 2.44% and 0.21% accuracy reductions, respectively, in the MNIST evaluation. *SignDefence* maintains robust performance with minimal impact, while blockchain-based approaches experience a 13.99% performance loss. In the HAR dataset, *SignMPC* shows improvements of 0.33%, 6.44%, 33.06%, and 12.08% over *SignDefence*, Flame, Flad, and BlockchainMPC techniques, respectively.

8.4 Chapter Summary

This chapter presented the *SignMPC* technique, a novel FL defence mechanism that strategically integrates SMPC, DP, and HE with an efficient sign-based aggregation protocol. The comprehensive experimental evaluation demonstrates that *SignMPC* consistently outperforms both established and recent defence techniques across diverse attack scenarios, with performance improvements ranging from modest gains of 0.33% over advanced methods like *SignDefence* to substantial advantages of 33.06% over Flad in Label Flipping attacks and 81.82% over FoolsGold in similar scenarios

while ensuring secure exchange and aggregation. The superior performance of *SignMPC* under various adversarial conditions stems from its fundamental design philosophy, which prioritizes gradient sign information over traditional magnitude-based aggregation approaches. This paradigm shift enables the system to maintain robustness against attacks that manipulate gradient magnitudes while preserving the directional information crucial for model convergence. The adaptive Jaccard similarity-based weighting mechanism dynamically adjusts client contribution weights based on gradient sign consensus, providing inherent resistance to Byzantine attacks and data poisoning attempts that systematically corrupt training updates.

The sign-based consensus mechanism represents *SignMPC*'s core innovation, effectively identifying and mitigating the impact of corrupted updates through its adaptive thresholding approach. This mechanism demonstrates remarkable resilience when enhanced with cryptographic primitives, maintaining consistent performance without the typical privacy-utility trade-offs observed in conventional defence approaches. The integration of DP mechanisms, HE, and SMPC protocols preserves model accuracy while ensuring comprehensive privacy protection throughout the FL process.

The validation across both HAR and MNIST datasets confirms the effectiveness of the proposed approach in maintaining model utility under sophisticated adversarial conditions that specifically target the integrity. This chapter presents the *SignMPC* technique, which combines SMPC, DP, and HE with efficient sign-based aggregation. Experimental results show that *SignMPC* outperforms established and recent defence techniques under various attack scenarios. However, the MNIST evaluation of Trimmed-Mean attacks reveals certain limitations of the approach. The superior performance of *SignMPC* under diverse attacks can be attributed to its fundamental design principle that focuses on gradient sign information rather than magnitude-based aggregation.

Chapter 9

Conclusions and Future Works

9.1 Conclusions

This thesis has addressed the critical challenges faced by FL systems, particularly the vulnerabilities to poisoning attacks, privacy threats, and inefficient client selection mechanisms. The comprehensive defence framework developed through this work demonstrates significant advancements in securing FL environments while maintaining model performance and computational efficiency. The primary contributions of this thesis can be summarized as follows:

9.1.1 Multi-layered Defence Architecture

The introduction of *FedChallenger* represents a paradigm shift in FL defence mechanisms. By implementing a dual-layer architecture combining zero-trust challenge-response authentication with adaptive Trimmed-Mean aggregation, this approach achieved substantial improvements over state-of-the-art methods. The experimental evaluation demonstrates consistent accuracy improvements of 3-10% across MNIST, FMNIST, EMNIST, and CIFAR-10 datasets, with convergence speeds 1.1-2.2 times faster than baseline approaches, including Stake, Shap, Cluster, Krum, and FedAvg.

The effectiveness of *FedChallenger* is particularly evident in adversarial scenarios. Under 30% Label Flipping attacks on FMNIST, the technique achieved remarkable accuracy improvements of 8.2%, 47.8%, 88.4%, and 7.4% over Trimmed-Mean, Krum, FedAvg, and DUEL approaches, respectively. More importantly, the convergence performance showed 60%, 52%, 31%, and 27%

faster training compared to these baselines in 20% Label Flipping scenarios. Even under severe 40% model poisoning attacks on CIFAR-10, *FedChallenger* maintained substantial accuracy leads, outperforming comparative methods by factors of 1.24 to 6.7 times.

9.1.2 Intelligent Client Selection Framework

The development of *Fed-Reputed* addresses the long-standing challenges of client selection in FL environments. By leveraging device capability information and implementing a modified Bellman equation within a hierarchical DQN-based framework, this approach successfully differentiates between malicious and straggler clients while addressing cold-start problems inherent in traditional reputation systems. The experimental validation demonstrates the robustness of *Fed-Reputed* across various adversarial scenarios. In MNIST evaluation with 40% malicious nodes, the technique achieved 15.01%, 37.86%, and 8.57% higher global model accuracy compared to SCS, RS, and MADDPG approaches, respectively. Similarly, in scenarios with 40% straggler nodes, *Fed-Reputed* demonstrated 8.88%, 40.11%, and 19.3% accuracy improvements while maintaining 1.38, 1.72, and 1.31 times faster convergence rates. The performance gains were consistently replicated across the FMNIST dataset, validating the technique's dataset-independent effectiveness.

9.1.3 Sign-based Aggregation Strategies

The introduction of *SignDefence* and *Ada-Sign* techniques addresses the fundamental limitations of traditional aggregation methods, particularly the dying ReLU problem and sensitivity to hyperparameter variations. *SignDefence* employs sign direction strategies combined with LeakyReLU and Jaccard similarity for model weight estimation, demonstrating consistent performance improvements across diverse attack scenarios on the HAR dataset.

The *Ada-Sign* extension, on the other hand, incorporates DP mechanisms while maintaining the robustness of sign-based aggregation. The technique's dynamic thresholding mechanism and leaky weighted aggregation provide effective defence against multiple attack vectors while preserving competitive performance in benign scenarios. The experimental results show 3-20% accuracy improvements over state-of-the-art techniques, including Flod, Flad, and Flame, while providing enhanced privacy guarantees through adaptive DP mechanisms.

9.1.4 Comprehensive Privacy Protection

The development of *SignMPC* represents the culmination of this research's privacy-preserving objectives. By incorporating highly configurable SMPC, HE, and DP algorithms, this approach ensures comprehensive communication protection without significant performance bottlenecks. The technique demonstrates 4-17% accuracy gains over existing approaches while maintaining robust privacy guarantees.

The superior performance of *SignMPC* under different attacks can be attributed to its focus on gradient sign information rather than magnitude-based aggregation. The robust aggregation algorithm dynamically weights client contributions based on adaptive Jaccard similarity thresholds, providing inherent resistance to attacks that manipulate training labels. This mechanism enables *SignMPC* to maintain consistent performance even when enhanced with privacy-preserving techniques, avoiding the typical privacy-utility degradation observed in conventional defence approaches.

9.2 Future Works

This research establishes a strong foundation for securing FL, but the dynamic nature of ML and cybersecurity demands continuous innovation. Future work should broaden the evaluation of defence mechanisms beyond image and sensor data to textual domains, including natural language processing and large language model fine-tuning. A critical area for improvement lies in advanced privacy attack mitigation, particularly against sophisticated, combined attack vectors like HidAttack, which merges privacy and poisoning attacks. Optimizing communication and computational efficiency will also be crucial for real-world applications, especially on resource-constrained edge devices; this involves developing lightweight aggregation protocols and energy-efficient defence strategies.

Further research must address scalability and heterogeneity challenges in large-scale FL deployments, involving thousands of diverse participants and various platforms. This includes developing distributed reputation systems and adaptive client selection strategies. Finally, it's vital that future research proactively considers regulatory and ethical implications. This means ensuring compliance with evolving privacy regulations like GDPR and CCPA, and addressing potential biases introduced

by defence mechanisms to promote ethical AI practices and fairness within FL systems. These crucial areas of investigation will help ensure FL systems remain robust, efficient, and practical for widespread deployment in the face of emerging threats.

Bibliography

[1] Gorka Abad, Stjepan Picek, and Aitor Urbieta. “SoK: On the security & privacy in federated learning”. In: *arXiv preprint arXiv:2112.05423* (2021).

[2] Mehdi Salehi Heydar Abad et al. “Hierarchical federated learning across heterogeneous cellular networks”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 8866–8870.

[3] Martin Abadi et al. “Deep learning with differential privacy”. In: *CCS* (2016).

[4] Sawsan AbdulRahman et al. “A survey on federated learning: The journey from centralized to distributed on-site learning and beyond”. In: *IEEE Internet of Things Journal* 8.7 (2020), pp. 5476–5497.

[5] Sawsan AbdulRahman et al. “FedMCCS: Multicriteria client selection model for optimal IoT federated learning”. In: *IEEE Internet of Things Journal* 8.6 (2020), pp. 4723–4735.

[6] Mahmoud A Albreem et al. “Green Internet of Things (GIoT): applications, practices, awareness, and challenges”. In: *IEEE Access* 9 (2021), pp. 38833–38858.

[7] Yasir Ali et al. “An Optimal Two-Step Approach for Defense Against Poisoning Attacks in Federated Learning”. In: *IEEE Access* (2025).

[8] Dan Ben Ami, Kobi Cohen, and Qing Zhao. “Client selection for generalization in accelerated federated learning: A multi-armed bandit approach”. In: *arXiv preprint arXiv:2303.10373* (2023).

[9] Sebastien Andreina et al. “Baffle: Backdoor detection via feedback-based federated learning”. In: *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*. IEEE. 2021, pp. 852–863.

[10] Mohamad Arafeh et al. “Independent and identically distributed (iid) data assessment in federated learning”. In: *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE. 2022, pp. 293–298.

[11] Borja Balle, Gilles Barthe, and Marco Gaboardi. “Privacy amplification by subsampling: Tight analyses via couplings and divergences”. In: *NeurIPS* (2018).

[12] Wugedele Bao et al. “Edge computing-based joint client selection and networking scheme for federated learning in vehicular IoT”. In: *China Communications* 18.6 (2021), pp. 39–52.

[13] Marco Barreno et al. “The security of machine learning”. In: *Machine Learning* 81 (2010), pp. 121–148.

[14] Enrique Tomás Martínez Beltrán et al. “Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges”. In: *IEEE Communications Surveys & Tutorials* (2023).

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

[16] Keith Bonawitz et al. “Practical secure aggregation for federated learning on user-held data”. In: *arXiv preprint arXiv:1611.04482* (2016).

[17] Nader Bouacida and Prasant Mohapatra. “Vulnerabilities in federated learning”. In: *IEEE Access* 9 (2021), pp. 63229–63249.

[18] Diego De Siqueira Braga et al. “Survey on computational trust and reputation models”. In: *ACM Computing Surveys (CSUR)* 51.5 (2018), pp. 1–40.

[19] David Byrd and Antigoni Polychroniadou. “Differentially private secure multi-party computation for federated learning in financial applications”. In: *Proceedings of the First ACM International Conference on AI in Finance*. 2020, pp. 1–9.

[20] Di Cao et al. “Understanding distributed poisoning attack in federated learning”. In: *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE. 2019, pp. 233–239.

[21] Xiaoyu Cao et al. “Fedrecover: Recovering from poisoning attacks in federated learning using historical information”. In: *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2023, pp. 1366–1383.

[22] Xiaoyu Cao et al. “Fltrust: Byzantine-robust federated learning via trust bootstrapping”. In: *arXiv preprint arXiv:2012.13995* (2020).

[23] Wenlin Chen, Samuel Horvath, and Peter Richtarik. “Optimal client sampling for federated learning”. In: *arXiv preprint arXiv:2010.13723* (2020).

[24] Xiao Chen, Chao Feng, and Shaohua Wang. “AIDFL: An Information-Driven Anomaly Detector for Data Poisoning in Decentralized Federated Learning”. In: *IEEE Access* (2025).

[25] Yu Chen et al. “A training-integrity privacy-preserving federated learning scheme with trusted execution environment”. In: *Information Sciences* 522 (2020), pp. 69–79.

[26] Jung Hee Cheon et al. “Homomorphic encryption for arithmetic of approximate numbers”. In: *Advances in cryptology–ASIACRYPT 2017: 23rd international conference on the theory and applications of cryptology and information security, Hong kong, China, December 3-7, 2017, proceedings, part i 23*. Springer. 2017, pp. 409–437.

[27] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. “Client selection in federated learning: Convergence analysis and power-of-choice selection strategies”. In: *arXiv preprint arXiv:2010.01243* (2020).

[28] Antonia Creswell et al. “Generative adversarial networks: An overview”. In: *IEEE signal processing magazine* 35.1 (2018), pp. 53–65.

[29] Ludwig Danwitz et al. “Parameter and model recovery of reinforcement learning models for restless bandit problems”. In: *Computational Brain & Behavior* 5.4 (2022), pp. 547–563.

[30] Ronald Doku and Danda B Rawat. “Mitigating data poisoning attacks on a federated learning-edge computing network”. In: *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE. 2021, pp. 1–6.

[31] Nanqing Dong et al. “Defending against poisoning attacks in federated learning with blockchain”. In: *IEEE Transactions on Artificial Intelligence* (2024).

[32] Ye Dong et al. “FLOD: Oblivious defender for private Byzantine-robust federated learning with dishonest-majority”. In: *European Symposium on Research in Computer Security*. Springer. 2021, pp. 497–518.

[33] Yarkın Doröz, Yin Hu, and Berk Sunar. “Homomorphic AES evaluation using the modified LTV scheme”. In: *Designs, Codes and Cryptography* 80 (2016), pp. 333–358.

[34] Minghong Fang et al. “Local model poisoning attacks to byzantine-robust federated learning”. In: *Proceedings of the 29th USENIX Conference on Security Symposium*. 2020, pp. 1623–1640.

[35] Z. Feng et al. “Securegbm: secure multi-party gradient boosting”. In: *2019 IEEE International Conference on Big Data (Big Data)* (2019). DOI: 10.1109/bigdata47090.2019.9006000.

[36] Hossein Fereidooni et al. “SAFELearn: Secure aggregation for private federated learning”. In: *2021 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2021, pp. 56–62.

[37] Yann Fraboni et al. “Clustered sampling: Low-variance and improved representativity for clients selection in federated learning”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 3407–3416.

[38] Yann Fraboni et al. “On the impact of client sampling on federated learning convergence”. In: (2021).

[39] Lei Fu et al. “Client selection in federated learning: Principles, challenges, and opportunities”. In: *IEEE Internet of Things Journal* (2023).

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

[41] Till Gehlhar et al. “SafeFL: MPC-friendly framework for private and robust federated learning”. In: *2023 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2023, pp. 69–76.

[42] Zhipin Gu and Yuexiang Yang. “Detecting malicious model updates from federated learning on conditional variational autoencoder”. In: *2021 IEEE international parallel and distributed processing symposium (IPDPS)*. IEEE. 2021, pp. 671–680.

[43] Prajjwal Gupta et al. “A Novel Data Poisoning Attack in Federated Learning based on Inverted Loss Function”. In: *Computers & Security* 130 (2023), p. 103270.

[44] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. “NTRU: A ring-based public key cryptosystem”. In: *International algorithmic number theory symposium*. Springer. 1998, pp. 267–288.

[45] Md Tamjid Hossain et al. “Desmp: Differential privacy-exploited stealthy model poisoning attacks in federated learning”. In: *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE. 2021, pp. 167–174.

[46] David C Howell. “Median absolute deviation”. In: *Encyclopedia of statistics in behavioral science* (2005).

[47] Tiansheng Huang et al. “Stochastic client selection for federated learning with volatile clients”. In: *IEEE Internet of Things Journal* 9.20 (2022), pp. 20055–20070.

[48] Matthew Jagielski et al. “Manipulating machine learning: Poisoning attacks and countermeasures for regression learning”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 19–35.

[49] Tayyebeh Jahani-Nezhad, Mohammad Ali Maddah-Ali, and Giuseppe Caire. “Byzantine-Resistant Secure Aggregation for Federated Learning Based on Coded Computing and Vector Commitment”. In: *arXiv preprint arXiv:2302.09913* (2023).

[50] Najeeb Moharram Jebreel et al. “Defending against the label-flipping attack in federated learning”. In: *arXiv preprint arXiv:2207.01982* (2022).

[51] J Jithish et al. “Distributed anomaly detection in smart grids: a federated learning-based approach”. In: *IEEE Access* 11 (2023), pp. 7157–7179.

[52] Aditya Pribadi Kalapaaking, Ibrahim Khalil, and Xun Yi. “Blockchain-based federated learning with SMPC model verification against poisoning attack for healthcare systems”. In: *IEEE Transactions on Emerging Topics in Computing* 12.1 (2023), pp. 269–280.

[53] Aditya Pribadi Kalapaaking et al. “Smpc-based federated learning for 6g-enabled internet of medical things”. In: *IEEE Network* 36.4 (2022), pp. 182–189.

[54] Keyvan Kazemi, Mohammad Hossein Badiei, and Hamed Kebriaei. “Robust Peer-to-Peer Federated Learning with Deep Reinforcement Learning Based Client Selection Against Data Poisoning Attacks”. In: *IEEE Transactions on Artificial Intelligence* (2025).

[55] Raouf Kerkouche, Gergely Ács, and Claude Castelluccia. “Federated learning in adversarial settings”. In: *arXiv preprint arXiv:2010.07808* (2020).

[56] Denise-Phi Khuu et al. “Data poisoning detection in federated learning”. In: *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*. 2024, pp. 1549–1558.

[57] Anastasiia Koloskova, Sebastian U Stich, and Martin Jaggi. “Sharper convergence guarantees for asynchronous SGD for distributed and federated learning”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 17202–17215.

[58] Satpal Singh Kushwaha et al. “Ethereum smart contract analysis tools: A systematic review”. In: *Ieee Access* 10 (2022), pp. 57037–57062.

[59] D. Li et al. “A nearest neighbor under-sampling strategy for vertical federated learning in financial domain”. In: *Proceedings of the 2022 ACM Workshop on Information Hiding and Multimedia Security* (2022). DOI: 10.1145/3531536.3532960.

[60] Jingtao Li et al. “Ressfl: A resistance transfer framework for defending model inversion attack in split federated learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10194–10202.

[61] Q. Li, Z. Wen, and B. He. “Practical federated gradient boosting decision trees”. In: (2019). DOI: 10.48550/arxiv.1911.04206.

[62] Xiaoxiao Li, Zhao Song, and Jiaming Yang. “Federated adversarial learning: A framework with convergence analysis”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 19932–19959.

[63] Zengpeng Li, Vishal Sharma, and Saraju P Mohanty. “Preserving data privacy via federated learning: Challenges and solutions”. In: *IEEE Consumer Electronics Magazine* 9.3 (2020), pp. 8–16.

[64] Zonghang Li et al. “Byzantine resistant secure blockchained federated learning at the edge”. In: *Ieee Network* 35.4 (2021), pp. 295–301.

[65] Enlu Lin, Qiong Chen, and Xiaoming Qi. “Deep reinforcement learning for imbalanced classification”. In: *Applied Intelligence* 50.8 (2020), pp. 2488–2502.

[66] Bo Liu et al. “When machine learning meets privacy: A survey and outlook”. In: *ACM Computing Surveys (CSUR)* 54.2 (2021), pp. 1–36.

[67] Xiaoyuan Liu et al. “Privacy-enhanced federated learning against poisoning adversaries”. In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 4574–4588.

[68] Yang Liu et al. “A secure federated transfer learning framework”. In: *IEEE Intelligent Systems* 35.4 (2020), pp. 70–82.

[69] Zizhen Liu et al. “DHSA: efficient doubly homomorphic secure aggregation for cross-silo federated learning”. In: *The Journal of Supercomputing* 79.3 (2023), pp. 2819–2849.

[70] Yunfei Long et al. “Fedcd: A classifier debiased federated learning framework for non-iid data”. In: *Proceedings of the 31st ACM International Conference on Multimedia*. 2023, pp. 8994–9002.

[71] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. “On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption”. In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 2012, pp. 1219–1234.

[72] Bing Luo et al. “Tackling system and statistical heterogeneity for federated learning with adaptive client sampling”. In: *IEEE INFOCOM 2022-IEEE conference on computer communications*. IEEE. 2022, pp. 1739–1748.

[73] Lingjuan Lyu et al. “Privacy and robustness in federated learning: Attacks and defenses”. In: *IEEE transactions on neural networks and learning systems* (2022).

[74] Jing Ma et al. “Privacy-preserving federated learning based on multi-key homomorphic encryption”. In: *International Journal of Intelligent Systems* 37.9 (2022), pp. 5880–5901.

[75] Abbass Madi et al. “A secure federated learning framework using homomorphic encryption and verifiable computing”. In: *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*. IEEE. 2021, pp. 1–8.

[76] Priyanka Mary Mammen. “Federated learning: Opportunities and challenges”. In: *arXiv preprint arXiv:2101.05428* (2021).

[77] Noora Al-Maslamani, Mohamed Abdallah, and Bekir Sait Ciftler. “Reputation-aware multi-agent DRL for secure hierarchical federated learning in IoT”. In: *IEEE Open Journal of the Communications Society* (2023).

[78] Ankita Maurya et al. “Federated Learning for Privacy-Preserving Severity Classification in Healthcare: A Secure Edge-Aggregated Approach”. In: *IEEE Access* (2025).

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

[80] H Brendan McMahan et al. “Learning differentially private recurrent language models”. In: *arXiv preprint arXiv:1710.06963* (2017).

[81] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. “The hidden vulnerability of distributed learning in byzantium”. In: *arXiv preprint arXiv:1802.07927* (2018).

[82] MA Moyeen et al. “FedChallenger: Challenge-Response-Based Defence for Federated Learning Against Byzantine Attacks”. In: *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE. 2023, pp. 3843–3848.

[83] MA Moyeen et al. “SignDefence: Byzantine-Robust Federated Learning with Sign Direction and Leaky ReLU”. In: *ICC 2025-2025 IEEE International Conference on Communications*. IEEE. 2025.

[84] Shijie Na, Yuzhi Liang, and Siu-Ming Yiu. “GPFL: A Gradient Projection-Based Client Selection Framework for Efficient Federated Learning”. In: *arXiv preprint arXiv:2403.17833* (2024).

[85] Dinh C Nguyen et al. “Federated learning for internet of things: A comprehensive survey”. In: *IEEE communications surveys & tutorials* 23.3 (2021), pp. 1622–1658.

[86] Thien Duc Nguyen et al. “{FLAME}: Taming backdoors in federated learning”. In: *31st USENIX Security Symposium (USENIX Security 22)*. 2022, pp. 1415–1432.

[87] Rajiv Kumar Pathni. “Artificial Intelligence and the Myth of Objectivity: Need for Regulation of Artificial Intelligence in Healthcare”. In: *Journal of Healthcare Management Standards (JHMS)* 3.1 (2023), pp. 1–14.

[88] Segun I Popoola et al. “Federated Deep Learning for Intrusion Detection in Consumer-Centric Internet of Things”. In: *IEEE Transactions on Consumer Electronics* (2023).

[89] Attia Qammar, Jianguo Ding, and Huansheng Ning. “Federated learning attack surface: taxonomy, cyber defences, challenges, and future directions”. In: *Artificial Intelligence Review* (2022), pp. 1–38.

[90] Attia Qammar et al. “Securing federated learning with blockchain: a systematic literature review”. In: *Artificial Intelligence Review* 56.5 (2023), pp. 3951–3985.

[91] Sumit Rai, Arti Kumari, and Dilip K Prasad. “Client selection in federated learning under imperfections in environment”. In: *AI* 3.1 (2022), pp. 124–145.

[92] Miguel A Ramirez et al. “New data poison attacks on machine learning classifiers for mobile exfiltration”. In: *arXiv preprint arXiv:2210.11592* (2022).

[93] Yanli Ren et al. “PurifyFL: Non-Interactive Privacy-Preserving Federated Learning Against Poisoning Attacks Based on Single Server”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2025).

[94] Atif Rizwan et al. “Intelligent digital twin for federated learning in aiot networks”. In: *Internet of Things* 22 (2023), p. 100698.

[95] Gaith Rjoub et al. “Trust-augmented deep reinforcement learning for federated learning client selection”. In: *Information Systems Frontiers* (2022), pp. 1–18.

[96] W. Ruan et al. “Private, efficient, and accurate: protecting models trained by multi-party learning with differential privacy”. In: (2022). DOI: 10.48550/arxiv.2208.08662.

[97] Sohag Sarkar. “Demystifying Decentralized Storage-A Critical Building Block of Future of Internet or Web 3.0”. In: *Telecom Business Review* 17.1 (2024), p. 18.

[98] Reent Schlegel et al. “CodedPaddedFL and CodedSecAgg: Straggler mitigation and secure aggregation in federated learning”. In: *IEEE Transactions on Communications* (2023).

[99] Devansh Shah et al. “Adversarial training in communication constrained federated learning”. In: *arXiv preprint arXiv:2103.01319* (2021).

[100] Virat Shejwalkar and Amir Houmansadr. “Manipulating the byzantine: Optimizing model poisoning attacks and defences for federated learning”. In: *NDSS*. 2021.

[101] Haoran Shi et al. “MVFLS: multi-participant vertical federated learning based on secret sharing”. In: *The Federate Learning* (2022), pp. 1–9.

[102] Junyu Shi et al. “Challenges and approaches for mitigating byzantine attacks in federated learning”. In: *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE. 2022, pp. 139–146.

[103] Yuxin Shi, Han Yu, and Cyril Leung. “Towards fairness-aware federated learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2023).

[104] Kristina P Sinaga and Miin-Shen Yang. “Unsupervised K-means clustering algorithm”. In: *IEEE access* 8 (2020), pp. 80716–80727.

[105] Artificial Intelligence Market Size. “Share & Trends Analysis Report by Solution, By Technology (Deep Learning, Machine Learning, Natural Language Processing, Machine Vision), By End Use, By Region, And Segment Forecasts, 2021-2028”. In: *San Francisco: Grand View Research* (2021).

[106] Jinhyun So et al. “Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 8. 2023, pp. 9864–9873.

[107] Behnaz Soltani et al. “A survey on participant selection for federated learning in mobile networks”. In: *Proceedings of the 17th ACM Workshop on Mobility in the Evolving Internet Architecture*. 2022, pp. 19–24.

[108] Zhendong Song et al. “Reputation-based federated learning for secure wireless networks”. In: *IEEE Internet of Things Journal* 9.2 (2021), pp. 1212–1226.

[109] Ekanut Sotthiwat et al. “Partially encrypted multi-party computation for federated learning”. In: *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE. 2021, pp. 828–835.

[110] Dimitris Stripelis et al. “Secure neuroimaging analysis using federated learning with homomorphic encryption”. In: *17th International Symposium on Medical Information Processing and Analysis*. Vol. 12088. SPIE. 2021, pp. 351–359.

[111] Octavian Suciu et al. “When does machine learning FAIL? generalized transferability for evasion and poisoning attacks”. In: *27th USENIX Security Symposium (USENIX Security 18)*. 2018, pp. 1299–1316.

[112] Le Sun, Jing Tian, and Ghulam Muhammad. “FedKC: Personalized Federated Learning With Robustness Against Model Poisoning Attacks in the Metaverse for Consumer Health”. In: *IEEE Transactions on Consumer Electronics* (2024).

[113] Xavier Tan et al. “Reputation-aware federated learning client selection based on stochastic integer programming”. In: *IEEE Transactions on Big Data* (2022).

[114] Peng Tang et al. “FLAD: Byzantine-Robust Federated Learning Based on Gradient Feature Anomaly Detection”. In: *IEEE Transactions on Dependable and Secure Computing* (2025).

[115] Jakub Tětek. “Approximate triangle counting via sampling and fast matrix multiplication”. In: *arXiv preprint arXiv:2104.08501* (2021).

[116] Mark Towers et al. *Gymnasium*. Mar. 2023. DOI: 10.5281/zenodo.8127026. URL: https://zenodo.org/record/8127025 (visited on 07/08/2023).

[117] Brandon Tran, Jerry Li, and Aleksander Madry. "Spectral signatures in backdoor attacks". In: *Advances in neural information processing systems* 31 (2018).

[118] Stacey Truex et al. "LDP-Fed: Federated learning with local differential privacy". In: *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*. 2020, pp. 61–66.

[119] Wei Wan et al. "A four-pronged defense against byzantine attacks in federated learning". In: *Proceedings of the 31st ACM International Conference on Multimedia*. 2023, pp. 7394–7402.

[120] Ning Wang et al. "FLARE: defending federated learning against model poisoning attacks via latent space representations". In: *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. 2022, pp. 946–958.

[121] Ruijin Wang et al. "RPIFL: Reliable and Privacy-Preserving Federated Learning for the Internet of Things". In: *Journal of Network and Computer Applications* 221 (2024), p. 103768.

[122] Xiao Wang, Lina Ge, and Guifeng Zhang. "A Review of Client Selection Mechanisms in Heterogeneous Federated Learning". In: *International Conference on Intelligent Computing*. Springer. 2023, pp. 761–772.

[123] Xiaofei Wang et al. "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning". In: *Ieee Network* 33.5 (2019), pp. 156–165.

[124] Yuwei Wang and Burak Kantarci. "A novel reputation-aware client selection scheme for federated learning within mobile environments". In: *2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE. 2020, pp. 1–6.

[125] Osama Wehbi et al. "Fedmint: Intelligent bilateral client selection in federated learning with newcomer iot devices". In: *IEEE Internet of Things Journal* (2023).

[126] Kang Wei et al. “Federated learning with differential privacy: Algorithms and performance analysis”. In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 3454–3469.

[127] Febrianti Wibawa et al. “BFV-Based Homomorphic Encryption for Privacy-Preserving CNN Models”. In: *Cryptography* 6.3 (2022), p. 34.

[128] Jiajun Wu, Steve Drew, and Jiayu Zhou. “Fedle: Federated learning client selection with lifespan extension for edge IoT networks”. In: *ICC 2023-IEEE International Conference on Communications*. IEEE. 2023, pp. 985–990.

[129] Qiong Wu et al. “HiFlash: Communication-efficient hierarchical federated learning with adaptive staleness control and heterogeneity-aware client-edge association”. In: *IEEE Transactions on Parallel and Distributed Systems* 34.5 (2023), pp. 1560–1579.

[130] Yusen Wu et al. “Tolerating adversarial attacks and byzantine faults in distributed machine learning”. In: *2021 IEEE International Conference on Big Data (Big Data)*. IEEE. 2021, pp. 3380–3389.

[131] Geming Xia et al. “Poisoning attacks in federated learning: A survey”. In: *IEEE Access* 11 (2023), pp. 10708–10722.

[132] Qi Xia et al. “FABA: an algorithm for fast aggregation against byzantine attacks in distributed neural networks”. In: *IJCAI*. 2019.

[133] Haoran Xie et al. “Verifiable Federated Learning With Privacy-Preserving Data Aggregation for Consumer Electronics”. In: *IEEE Transactions on Consumer Electronics* (2023).

[134] Ao Xiong et al. “A truthful and reliable incentive mechanism for federated learning based on reputation mechanism and reverse auction”. In: *Electronics* 12.3 (2023), p. 517.

[135] Mengchu Xu et al. “FedAG: A Federated Learning Method Based on Data Importance Weighted Aggregation”. In: *2023 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE. 2023, pp. 1–6.

[136] R. Xu et al. “Detrust-fl: privacy-preserving federated learning in decentralized trust setting”. In: (2022). DOI: 10.48550/arxiv.2207.07779.

[137] Runhua Xu et al. “Detrust-FL: Privacy-preserving federated learning in decentralized trust setting”. In: *2022 IEEE 15th International Conference on Cloud Computing (CLOUD)*. IEEE. 2022, pp. 417–426.

[138] Yajing Xu et al. “Besifl: Blockchain empowered secure and incentive federated learning paradigm in iot”. In: *IEEE Internet of Things Journal* (2021).

[139] Zhuoqun Yan et al. “Comments on “VCD-FL: Verifiable, Collusion-Resistant, and Dynamic Federated Learning””. In: *IEEE Transactions on Information Forensics and Security* (2025).

[140] Li Yang et al. “Enhanced model poisoning attack and Multi-Strategy defense in federated learning”. In: *IEEE Transactions on Information Forensics and Security* (2025).

[141] Ming Yang et al. “Model poisoning attack in differential privacy-based federated learning”. In: *Information Sciences* 630 (2023), pp. 158–172.

[142] Junyu Ye et al. “A Client Detection and Parameter Correction Algorithm for Clustering Defense in Clustered Federated Learning”. In: *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 2024, pp. 2383–2388.

[143] Dong Yin et al. “Byzantine-robust distributed learning: Towards optimal statistical rates”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 5650–5659.

[144] Bin Yu et al. “A survey on federated learning in data mining”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 12.1 (2022), e1443.

[145] Menghang Yu et al. “A Comprehensive Study on Personalized Federated Learning with Non-IID Data”. In: *2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. IEEE. 2022, pp. 40–49.

[146] Ruihao Yuan et al. “A Variational Auto-Encoder Enabled Multi-Band Channel Prediction Scheme for Indoor Localization”. In: *ICC 2023-IEEE International Conference on Communications*. IEEE. 2023, pp. 571–576.

[147] Kai Yue et al. “Federated learning via plurality vote”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[148] Chengliang Zhang et al. “{BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning”. In: *2020 USENIX annual technical conference (USENIX ATC 20)*. 2020, pp. 493–506.

[149] Hangjia Zhang et al. “Adaptive client selection in resource constrained federated learning systems: A deep reinforcement learning approach”. In: *IEEE Access* 9 (2021), pp. 98423–98432.

[150] Li Zhang et al. “Homomorphic encryption-based privacy-preserving federated learning in IoT-enabled healthcare system”. In: *IEEE transactions on network science and engineering* 10.5 (2022), pp. 2864–2880.

[151] Sai Qian Zhang, Jieyu Lin, and Qi Zhang. “A multi-agent reinforcement learning approach for efficient client selection in federated learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 8. 2022, pp. 9091–9099.

[152] Tuo Zhang et al. “Federated learning for the internet of things: Applications, challenges, and opportunities”. In: *IEEE Internet of Things Magazine* 5.1 (2022), pp. 24–29.

[153] Ziqi Zhang et al. “FedSlice: Protecting Federated Learning Models from Malicious Participants with Model Slicing”. In: *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE. 2023, pp. 460–472.

[154] Ying Zhao et al. “PDGAN: A novel poisoning defense method in federated learning using generative adversarial network”. In: *Algorithms and Architectures for Parallel Processing: 19th International Conference, ICA3PP 2019, Melbourne, VIC, Australia, December 9–11, 2019, Proceedings, Part I 19*. Springer. 2020, pp. 595–609.

[155] Zihao Zhao, Mengen Luo, and Wenbo Ding. “Deep leakage from model in federated learning”. In: *arXiv preprint arXiv:2206.04887* (2022).

[156] Ian Zhou et al. “Secure multi-party computation for machine learning: A survey”. In: *IEEE Access* 12 (2024), pp. 53881–53899.

[157] Xingjie Zhou et al. “FLGuardian: Defending against Model Poisoning Attacks via Fine-grained Detection in Federated Learning”. In: *IEEE Transactions on Information Forensics and Security* (2025).

[158] Hangyu Zhu et al. “Federated learning on non-IID data: A survey”. In: *Neurocomputing* 465 (2021), pp. 371–390.

[159] Hongbin Zhu et al. “Online client selection for asynchronous federated learning with fairness consideration”. In: *IEEE Transactions on Wireless Communications* 22.4 (2022), pp. 2493–2506.

[160] Giulio Zizzo et al. “Fat: Federated adversarial training”. In: *arXiv preprint arXiv:2012.01791* (2020).

[161] Shaojun Zuo et al. “ApaPRFL: Robust Privacy-Preserving Federated Learning Scheme Against Poisoning Adversaries for Intelligent Devices Using Edge Computing”. In: *IEEE Transactions on Consumer Electronics* (2024).